

DISLOCATION MODELING OF ORIENTATION GRADIENTS IN PHASE-TRANSFORMED TANTALUM THIN FILMS

A Thesis

Presented to the Faculty of the Graduate School
of Cornell University
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

by

Ari Philip Kestenbaum

August 2016

© 2016 Ari Philip Kestenbaum
ALL RIGHTS RESERVED

ABSTRACT

Tantalum thin films are used in a wide range of applications, from micro- and nanoelectronics to military and aerospace applications, to medical devices. Tantalum has two phases, the stable bulk α phase and the metastable β phase found only in thin films. In a recent discovery, it has been shown that when tantalum is phase-transformed from β to α , it can form a unique microstructure with continuous changes in orientation and discontinuous grain boundaries. Dislocation structures must be present in order to account for the lattice curvature associated with orientation gradients. To identify the dislocation arrays that give rise to this microstructure, we first analyzed the electron backscatter diffraction (EBSD) data of the orientation gradients using geometrically necessary dislocations (GNDs), but we found that our EBSD data are too noisy and do not have sufficient resolution for this approach. In this work, two models are presented in an attempt to describe the dislocation structures that must be present to account for the orientation gradients. In the first, a method of generating smooth, noise-free orientation data that match key features of the actual phase-transformed microstructure is developed and validated by comparing model data with film data. In the second model, a genetic algorithm approach is used to generate dislocation structures that can account for the orientation gradients produced using the first model. The genetic algorithm model consistently generates dislocation structures that produce orientation maps with an average misorientation of less than 2° from the smooth orientation gradients generated by the first model. The generated dislocation structure is consistent across multiple runs, with similar shapes and Burgers vectors. The genetic algorithm model selects two types of dislocations, mostly ($\sim 80\%$) $[\bar{1}\bar{1}1]$ and the rest $[\bar{1}11]$. The dominance of these two slip systems provides a starting point to search for a mechanism in the β -to- α phase transformation that is capable of producing these dislocations.

Biographical Sketch

Ari Kestenbaum was born and raised in Westchester, NY just outside New York City. He attended Cornell University for his undergraduate education and received a B.S. in materials science with a minor in computer science. He liked Cornell so much he decided to stay for a fifth year.

Acknowledgements

First and foremost I want to thank my parents for their tremendous love and support. I would also like to thank my advisor, Shefford Baker, for his advice and guidance, as well as his assistance in seeing the bigger picture whenever I got too wrapped up in the details of whatever I was doing. I also want to thank him for giving me the opportunity to do research as an undergraduate and for giving me a real problem to work on. I also want to thank Michael Thompson, who manages to pack more into less time than anyone else I know; each conversation with him turned into months of research. I would also like to thank him for his suggestion to use a genetic algorithm for creating dislocation structures. Last, but certainly not least, I am very grateful to Betsy Ellis, who has been there to help me every time I needed it during the last three years of research.

Table of Contents

Biographical Sketch	iii
Acknowledgements	iv
Table of Contents	v
List of Tables	vi
List of Figures	vii
1 Introduction	1
2 Background	5
2.1 Tantalum Thin Films	5
2.2 Geometrically Necessary Dislocations	7
2.3 Genetic Algorithms	9
3 Methods	12
3.1 Plotting Orientation Maps	12
3.2 Geometrically Necessary Dislocations	14
3.3 The Smooth Model	15
3.4 The Genetic Algorithm Model	21
4 A Genetic Algorithm for Dislocation Structure Prediction in Phase-Transformed Tantalum Thin Films	26
4.1 Introduction	27
4.2 Geometrically Necessary Dislocations	31
4.2.1 GND Calculations	31
4.2.2 GND Calculation Results	32
4.3 Model Microstructure with Smooth Gradients	33
4.3.1 Model Development	34
4.3.2 Model Validation (Results and Discussion)	37
4.4 Genetic Algorithm Model	42
4.4.1 Details and Methods	44
4.4.1.1 Individual Encoding	45
4.4.1.2 Fitness Function	50
4.4.1.3 Initialization, Selection, Crossover, and Mutation	50
4.4.2 Model Validation (Results)	52
4.5 Discussion	57
4.6 Summary and Conclusions	65
5 Conclusions and Outlook	67
Bibliography	68

List of Tables

4.1	Comparison of properties of α and β tantalum. Data from [7] except where noted.	28
-----	--	----

List of Figures

1.1	EBSD data orientation maps of (a) an as-deposited α film with standard thin-film microstructure and (b) a phase-transformed α film [1], colored by out-of-plane orientation according to the inset stereographic triangle. Black and gray lines denote high- and low-angle boundaries (above 8° and between 4 and 8°) respectively, and black pixels indicate unindexed portions of the film. Unique features of the phase-transformed microstructure include long-range orientation gradients and discontinuous grain boundaries.	2
2.1	EBSD orientation maps of a series of phase-transformed films sputtered at argon pressures of 2, 8, 12, 14, 15, and 16 mTorr in (a), (b), (c), (d), (e), and (f) respectively [1]. The patterns and sizes of the orientation gradients change dramatically with the argon pressure.	7
2.2	Arrays of (a) edge and (b) screw dislocations, each causing an orientation gradient. Arrays of mixed dislocations can also result in orientation gradients.	8
3.1	Orientation data from the indicated regions of the film in Figure 2.1e. At each point the orientation is plotted with the OOP color and a vector pointing in the (001) direction in crystal (local) coordinates. This unusual pattern is an indication of a radial transformation process.	16
3.2	A simple radial rotation model, which starts in the center and rotates outward radially, rotating about an axis perpendicular to the radial direction. Both are colored by their OOP orientation component according to the inset stereographic triangle. (a) is the result from the simple model and (b) is the corresponding region of the film data. The match of symmetry is an indication that the phase transformation is a radial process.	17
3.3	A plot of the Equation (3.1), which broadens the (011) stripes and shrinks the (001) regions. Note that the profile is broader near -90° , 30° , and 150° , which correspond to (011) stripes on the triangle, and that it is narrower near -30° , 90° , and 210° , which correspond with the (001) regions at the corners of the triangle.	18
3.4	A plot of Equation (3.2). The value of this function gives the magnitude of rotation at each point. Note the threefold symmetry and the sigmoidal profile along any path extending radially from the origin.	19
3.5	A radial rotation model with the adjustments incorporated. The model data (a) now not only match the symmetry of the film data (b), but also the spatial distribution of orientations. The close match orientations is an indication that the phase transformation is a radial process.	20

4.1	Out-of-plane (OOP) orientation of a series of phase-transformed tantalum films, sputtered at argon pressures of 14, 15, and 16 mTorr for (a), (b), and (c) respectively [1]. Colors correspond to orientation given by the inset stereographic color triangle; black and gray lines denote high- and low-angle boundaries. Distinct features include continuous orientation gradients and discontinuous grain boundaries.	29
4.2	GND density maps for (a) $(101)[\bar{1}\bar{1}1]$ and (b) $(10\bar{1})[1\bar{1}1]$ slip systems for the film in Figure 4.1b. These are difficult to interpret primarily because the EBSD orientation data used in the calculations are noisy.	34
4.3	Film data with overlaid triangles indicating sections of the film that were modeled. Most of the modeling work was focused on Triangle A, but pole figures of the models of Triangles B and C are shown in Figures 4.9 and 4.10.	35
4.4	Illustration of phenomenological insight that matches triangular regions in the film. (a) is an illustration of a cube with a (111) OOP direction rotated one way resulting in a (001) OOP direction, and rotated another way resulting in a (011) OOP orientation. The cubes are colored by their OOP orientation by the stereographic triangle. (b) is the region of the film that this process matches	36
4.5	(a) Smooth model constant-rate radial rotation map, (b) section of film data from Figure 4.1b that was modeled. Both are colored by OOP orientation according to the inset stereographic triangle. Note the symmetry matches, indicating a radial process yields a good match.	37
4.6	Schematic for generating orientation data from smooth model in Figure 4.5. (a) selecting the point at which to calculate the orientation, and (b) illustration of the orientation calculation via rotation, where the cube represents the starting orientation. O is the origin; P is the point at which to compute the orientation; $vecr$ is the radial vector; \vec{v}_r is the axis of rotation, which is perpendicular to \vec{r} and lies in the plane; ω is the amount of rotation and is proportional to the distance between P and O	38
4.7	(a), (b), and (c) are components along the respective (001) , (100) , and (010) axes of film orientation data for the region of the film shown in Figure 4.5b, and are colored by the inset stereographic triangle. (d), (e), and (f) are the model data for the same region along the same respective axes. The black pixels in (a–c) are unindexed portions of the film.	39
4.8	Pole figures of film and model data from Triangle A in Figure 4.3. (a–c) are film data and (d–f) are model data. (a) and (d) are (001) pole figures, (b) and (e) are (011) pole figures, and (c) and (f) are (111) pole figures.	40
4.9	Pole figures of film and model data from Triangle B in Figure 4.3. (a–c) are film data and (d–f) are model data. (a) and (d) are (001) pole figures, (b) and (e) are (011) pole figures, and (c) and (f) are (111) pole figures.	41
4.10	Pole figures of film and model data from Triangle C in Figure 4.3. (a–c) are film data and (d–f) are model data. (a) and (d) are (001) pole figures, (b) and (e) are (011) pole figures, and (c) and (f) are (111) pole figures.	42

4.11	GND density maps for (a) $(101)[\bar{1}\bar{1}1]$ and (b) $(10\bar{1})[1\bar{1}1]$ slip systems for the model triangle. Using the smooth model data has removed the noise in Figure 4.2, but still does not yield a useful GND density map.	43
4.12	Due to its threefold rotational symmetry, this one-sixth of the smooth model triangle is all that needs to be modeled, allowing more detail and reducing computation time.	45
4.13	(a) 3D and (b) 2D views of the contour lines from the level set function with no noise added; (c) 2D view of contour lines with noise added. The use of contour lines to represent dislocations ensures that they do not cross while still allowing for their shape to change.	47
4.14	Orientation comparison between the smooth model and the genetic algorithm. (a) is the orientation data from the smooth model, (b) is orientation data from the genetic algorithm, (c) is the misorientation between (a) and (b), and (c) and (d) are the in-plane and out-of-plane dislocations, respectively, colored by Burgers vector. This individual has a fitness metric of 4.84, an average misorientation of 1.39° , and a dislocation density of $3.45 \times 10^{14} \text{ m}^{-2}$	54
4.15	Results from several different runs of the algorithm; each is the best individual in its run. For each result, the first image is the orientation from the smooth model, the second image is the orientation map that is generated from the set of dislocations of an individual, the third is the misorientation between the first two, and the third and fourth are IP and OOP dislocations, respectively, colored by Burgers vector. (a–f) is the same as Figure 4.14. (f–j) has fitness metric 4.80, average misorientation 1.59° , dislocation density $3.21 \times 10^{14} \text{ m}^{-2}$. (k–o) has fitness metric 4.89, average misorientation 1.61° , dislocation density $3.28 \times 10^{14} \text{ m}^{-2}$. The algorithm converges on a similar set of IP dislocations each run, with the majority ($\sim 80\%$) $[\bar{1}\bar{1}1]$ dislocations and almost all of the rest $[\bar{1}11]$ dislocations. The different types of dislocations converge to similar areas of the triangles, as well as similar shapes	55
4.16	Multi-generation statistics plot for a single run of the genetic algorithm. This tracks the following metrics from the best individual from each generation: (a) average misorientation and dislocation density; (b) the number of noise points, the number of IP dislocations, and the number of OOP dislocations.	56
4.17	Orientation comparison between the smooth model and the genetic algorithm using only $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors. (a) is the orientation data from the smooth model, (b) is orientation data from the genetic algorithm, (c) is the misorientation between (a) and (b), and (c) and (d) are the in-plane and out-of-plane dislocations, respectively, colored by Burgers vector. This individual has a fitness metric of 5.08, an average misorientation of 1.36° , and a dislocation density of $3.72 \times 10^{14} \text{ m}^{-2}$. The low misorientation indicates that the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ are sufficient for the genetic algorithm model to match the orientation data of the smooth model.	59

- 4.18 Orientation comparison between the smooth model and the genetic algorithm, run using all $\langle 111 \rangle$ -type Burgers vectors except $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$. (a) is the orientation data from the smooth model, (b) is orientation data from the genetic algorithm, (c) is the misorientation between (a) and (b), and (c) and (d) are the in-plane and out-of-plane dislocations, respectively, colored by Burgers vector. This individual has a fitness metric of 13.64, an average misorientation of 11.1° , and a dislocation density of $2.54 \times 10^{14} \text{ m}^{-2}$. The high misorientation indicates that the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors are necessary for the genetic algorithm model to match the orientation data of the smooth model. 60
- 4.19 A plot of the pixel-to-pixel misorientation gradient for (a) the smooth model and (b) the individual in Figure 4.14, both colored by the same scale bar. This is a plot of the misorientation between adjacent pixels divided by the distance between them. For (a) this is in the x direction and for (b) this is in the tangential direction, perpendicular to the radial direction. The same regions of both triangles have the highest misorientation but it is not clear why the misorientation gradients are so much higher in (b) than in (a). . . . 61

Chapter 1:

Introduction

Thin films are an integral part of modern technology and are important in, for example, microelectronic, optical, and medical devices. Thin films can range from just a monolayer up to several micrometers in thickness, and are typically deposited on rigid substrates. Their properties can be very different from those of the same material in bulk, and are highly dependent on deposition and processing conditions. Thin films are often chosen for their mechanical, chemical, or electronic properties, all of which depend on the microstructure of the film. However, thin films often have a non-equilibrium microstructure, which makes their stability important for device operation. Thus the relationship between microstructure and properties of thin films is crucial to device reliability.

Tantalum thin films are used in a wide variety of applications, including thin-film capacitors, x-ray lithography masks, and diffusion barriers between copper and silicon. One reason tantalum is used in such a range of applications is because it can be deposited as a thin film in one of two phases, the stable bulk phase α or the metastable thin-film-only β phase, which have very different mechanical and electronic properties. This makes it important to be able to select which phase is deposited, but depositing the desired phase can be difficult. Additionally, since the β phase is metastable, preventing it from transforming to the α phase during deposition, processing, or device operation is an important consideration because the phase transformation can begin at several hundred degrees Celsius. The phase transformation also causes a large change in stress in the film, which, beyond being detrimental to device performance, can lead to device failure.

The microstructure that arises from this phase transformation is unique, with features not typically associated with crystalline materials. The contrast between the standard thin-film microstructure and that of phase-transformed tantalum is shown in Figure 1.1, with

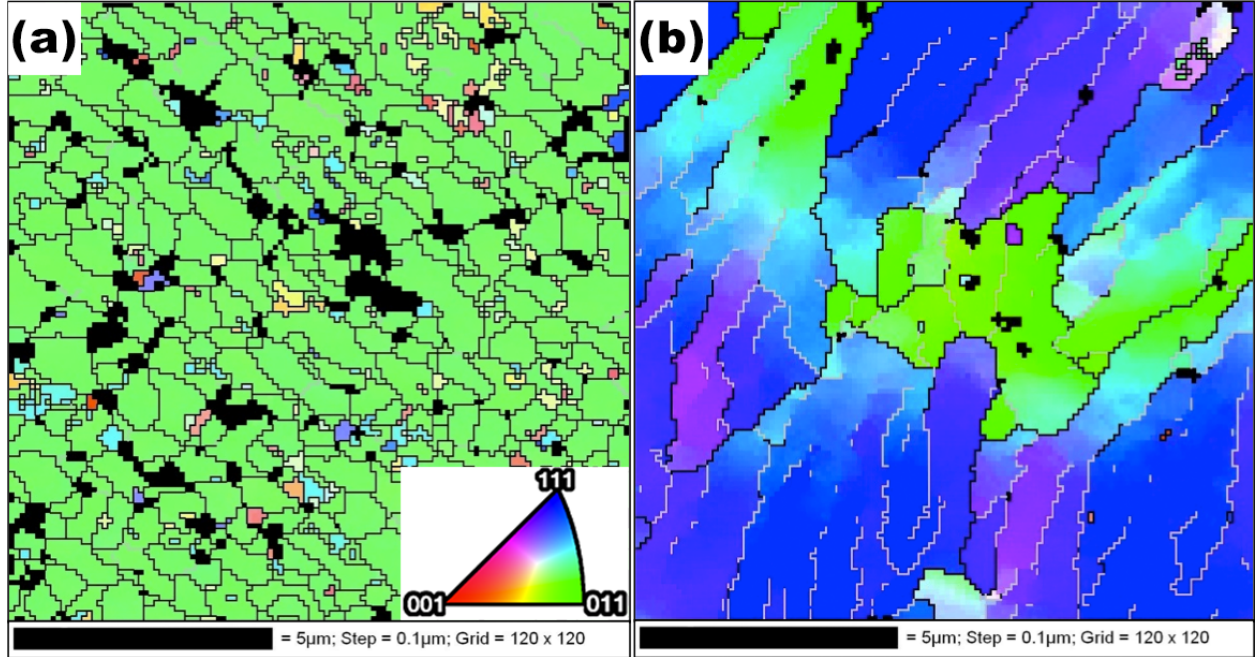


Figure 1.1: EBSD data orientation maps of (a) an as-deposited α film with standard thin-film microstructure and (b) a phase-transformed α film [1], colored by out-of-plane orientation according to the inset stereographic triangle. Black and gray lines denote high- and low-angle boundaries (above 8° and between 4 and 8°) respectively, and black pixels indicate unindexed portions of the film. Unique features of the phase-transformed microstructure include long-range orientation gradients and discontinuous grain boundaries.

maps of EBSD orientation data colored by their out-of-plane (OOP) orientation according to the inset stereographic triangle. Figure 1.1a is an as-deposited α film with standard thin-film microstructure and Figure 1.1b is a phase-transformed film [1]. Black and gray lines are typically called high- and low-angle grain boundaries, but because it is so difficult to determine what exactly a grain would be in this microstructure, they will be referred to as orientation boundaries. Black lines denote high-angle orientation boundaries, those above 8° , and gray lines denote low-angle orientation boundaries, those between 4 and 8° . The as-deposited α film has strong (110) texture and an average grain size on the order of the film thickness, approximately 400 nm [1]. The phase-transformed film exhibits large, long-range orientation gradients, with orientations that can change by dozens of degrees without crossing any high- or low-angle orientation boundaries. Some of the high-angle orientation boundaries transition to low-angle orientation boundaries and simply disappear, which makes it possible to move from one side of a boundary around to the other without

crossing it. High-angle orientation boundaries are much farther apart than the film thickness, which is approximately $1.8\text{ }\mu\text{m}$ [1].

The only way to account for such orientation gradients in a crystalline material is with dislocations, and so it is necessary to understand the dislocation structure produced by the phase transformation in order to better understand the phase transformation itself. The only tool that has been used in literature to analyze orientation gradients in terms of dislocations is geometrically necessary dislocation (GND) density. We calculated the GND density in our films, but as discussed in Section 4.2 we found that the electron backscatter diffraction (EBSD) orientation data are too noisy for this approach. This is because the noise (the uncertainty in the orientation of any one pixel) is similar to the orientation gradient (the average change in orientation from one pixel to the next). We then developed a model that generates smooth, noise-free orientation data that match key aspects of the phase-transformed microstructure. However, as discussed in Section 4.3 we found that the GND analysis does not work on this smooth data either, largely because the orientation data are now artificial and so dislocations in them wouldn't be constrained the way they would in a physical system. Using this smooth orientation data we then took the opposite approach; instead of attempting to determine directly from the orientation data what dislocations could account for it, we developed a model that produces dislocation structures, computes how well the resulting orientation data match the smooth model data, and then modifies the dislocation structure to better match the smooth model data. We used a genetic algorithm for this model because genetic algorithms work well for this type of iterative approach.

Genetic algorithms (GAs) are a subfield of evolutionary computation, and are an optimization technique or metaheuristic that mimics the process of evolution. GAs have been successful in a tremendous number of fields, and offer advantages over other optimization techniques in many circumstances, including better performance with complex or unintuitive optimization criteria and less premature convergence. They are thus better at approaching the globally optimal solution, and they place few restrictions on what they are optimiz-

ing, allowing functions that are not smooth or even continuous to be optimized, which is particularly important when computing discrete entities such as dislocations.

This thesis is divided into five chapters. This chapter provided motivation and a brief problem statement for this research. Chapter 2 provides background information on tantalum thin films, geometrically necessary dislocations, and genetic algorithms, and describes the problem statement in more detail. Chapter 3 describes the geometrically necessary dislocation calculations that were used, and provides details on the methods used in developing both the smooth model and the genetic algorithm model. Chapter 4 presents the results and discussion for both models and is under preparation for journal submission. Chapter 5 summarizes the conclusions of this work and presents suggestions for further work.

Chapter 2:

Background

2.1 Tantalum Thin Films

Tantalum films are important in many microelectronic and medical device applications. Tantalum is very dense ($\sim 16.6 \text{ g/cm}^3$) [2] with a very high melting point (3269 K) [3] and absorbs x-rays well [4]. It forms a thin oxide [5] which gives it excellent corrosion resistance [6].

Tantalum has two phases with very different properties, and deposition conditions have a large effect on which phase is deposited. The α phase is the equilibrium bulk phase, has a BCC structure with space group $Im\bar{3}m$, and is ductile [7]. It is nearly immiscible in copper [8], so it is used as a diffusion barrier between copper and silicon in microelectronic devices [9–11]. It also has low resistivity ($15\text{--}60 \text{ }\mu\Omega\cdot\text{cm}$) [7] so it is often used in thin film capacitors [12,13]. The β phase, first discovered by Read and Altman in 1965 [14], by contrast is only found in thin films [15], has a complicated tetragonal σ -type Frank-Kasper structure with space group $P\bar{4}2_1m$ and a 30-atom unit cell [16], is brittle [17], and is calculated to be slightly less dense than the α phase ($\sim 16.3 \text{ g/cm}^3$) [7]. It is more reactive with chlorine-based plasmas than the α phase [18,19], which combined with tantalum's x-ray absorption makes it a suitable material for x-ray lithography masks [20]. It has a high resistivity ($170\text{--}210 \text{ }\mu\Omega\cdot\text{cm}$) [7] and a low temperature coefficient of resistance ($(-150 \pm 30) \times 10^{-6} \text{ K}^{-1}$) [13] which make it a good material for thin-film resistors [12,13]. The β phase is also metastable, and will transform to α when heated [21–23].

The β -to- α phase transformation has been reported widely but rarely studied systematically, leading to very different results because the transformation process is highly dependent on the details of deposition, annealing, and substrate. The transformation temperature has

been reported to start as low as 300°C in some cases [23, 24] and to not finish transforming until 800°C in others [25], partially because the phase transformation has been shown to be very sensitive to the amount of oxygen in the atmosphere [24, 26]. The phase transformation also causes a large change in stress in the tensile direction [24, 27, 28] if the annealing environment is oxygen-free, or in the compressive direction if the environment is not [29, 30]. See Knepper *et al.* [24, 28, 31] and Baker *et al.* [1, 32, 33] for more details about the phase transformation as well as the effects of deposition conditions on phase formation.

The phase transformation of β to α tantalum produces a unique microstructure with highly unusual features, especially for a crystalline material. The canonical thin-film microstructure is columnar, and consists of well-defined grains, each having a single orientation, usually with an average size on the order of the film thickness, separated by continuous high-angle grain boundaries [34]. In contrast, the microstructure of the phase-transformed films consists of large long-range orientation gradients, some as high as 4 or 5°/ μm that extend for tens of micrometers, producing total misorientations of dozens of degrees. This microstructure also contains discontinuous grain boundaries, with low-angle boundaries starting in the middle of the film, transitioning to a high-angle boundary, then back to a low-angle boundary, and ending in another part of the film. This makes it difficult to determine what actually constitutes a ‘grain’ in the phase-transformed microstructure.

Figure 2.1 shows a series of phase-transformed films [1] that were sputtered over a range of argon pressures, from 2 mTorr (Figure 2.1a) to 16 mTorr (Figure 2.1f). There is a strong dependence of the resulting phase-transformed microstructure on the argon pressure. The microstructures range from Figure 2.1a, which has the shortest orientation gradients and smallest grains, of a few μm , all the way to Figure 2.1f, which has orientation gradients so long and grains so large they are difficult to see with a window size of approximately 55 $\mu\text{m} \times 50 \mu\text{m}$. The films were all 400–500 nm thick [1], so the ‘grains’ in 2.1f that extend for dozens of μm are noteworthy. The most interesting sample is Figure 2.1e because the others either have grains too small to see orientation gradients of more than a few μm (figs. 2.1a

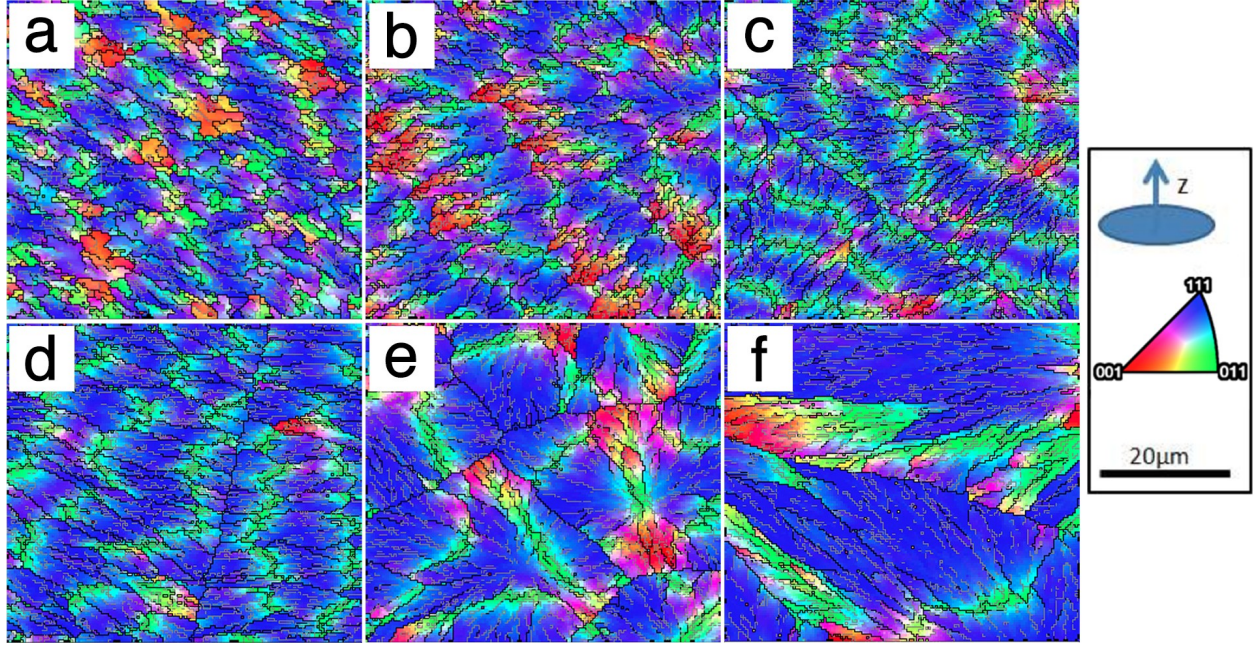


Figure 2.1: EBSD orientation maps of a series of phase-transformed films sputtered at argon pressures of 2, 8, 12, 14, 15, and 16 mTorr in (a), (b), (c), (d), (e), and (f) respectively [1]. The patterns and sizes of the orientation gradients change dramatically with the argon pressure.

to 2.1d), or orientation gradients so large that they are difficult to see on the orientation map (fig. 2.1f). Figure 2.1e has sufficient alignment of grain size, orientation gradient magnitude and length, and window size to be able to see substantial orientation gradients in multiple places. Some of the gradients in Figure 2.1e transition from (1 1 1) to (0 1 1) to (0 0 1) without crossing any high- or low-angle orientation boundaries, a total change of approximately 80° over approximately a dozen μm .

2.2 Geometrically Necessary Dislocations

The only way to account for orientation gradients of this magnitude in a pure, crystalline material is with dislocations. A common approach is to calculate geometrically necessary dislocations (GNDs), which are the minimum set of (unpaired) dislocations that are necessary to account for observed lattice curvature. The concept was first introduced by Nye [35], who described a geometric relation between lattice curvature and the density of unpaired dislocations. A simple example of lattice curvature arising from unpaired dislocations is

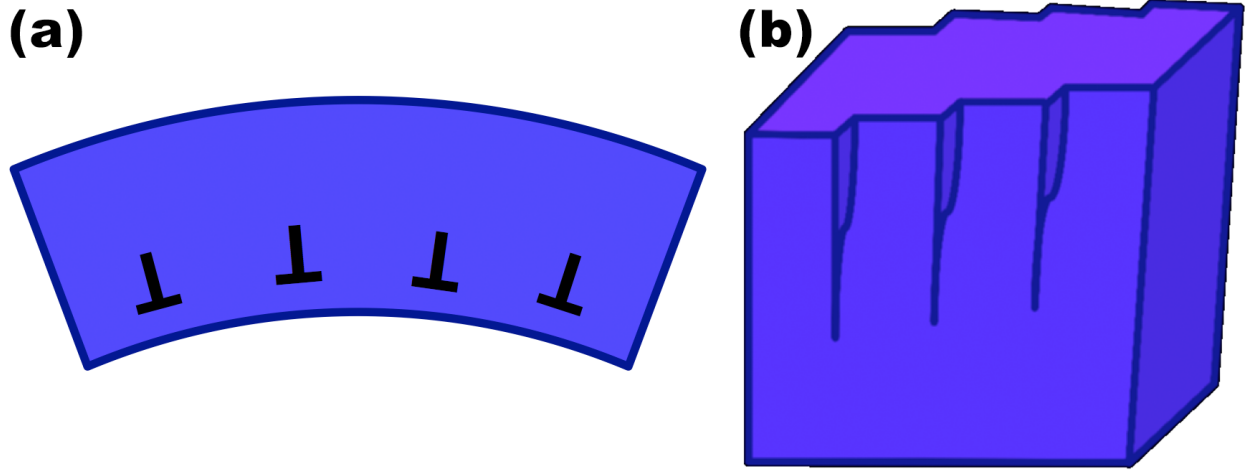


Figure 2.2: Arrays of (a) edge and (b) screw dislocations, each causing an orientation gradient. Arrays of mixed dislocations can also result in orientation gradients.

shown in Figure 2.2, where Figure 2.2a shows an array of edge dislocations that results in an orientation gradient and Figure 2.2b shows a similar result for an array of screw dislocations. An array of mixed dislocations may cause an orientation gradient as well. Dislocation arrays only cause orientation gradients if their Burgers and sense vectors are aligned properly, otherwise they can cancel and will no longer contribute to lattice curvature.

The idea of GNDs, first formulated as a tensor describing the ‘state of dislocation’ at a point, was introduced by Nye [35] and extended by Kröner [36,37] and Ashby [38]. Important contributions to the GND framework also include those by Arsenlis and Parks [39], Sun *et al.* [40], El-Dasher *et al.* [41], and Kysar *et al.* [42]. The GND framework has been applied to both bulk materials [40–45] and thin films [46–49] as a way to understand strain gradient plasticity [50–53]. Orientation gradients have been observed in many bulk samples [54–59] and some thin films [60–63], sometimes reported with GND analysis [57–60] and sometimes not [55–57,61–63]. GND analysis performed on samples with orientation gradients is generally concerned with the effect of GNDs on hardness and yield stress, as nearly all of these samples have been heavily deformed or highly strained. The only previously observed orientation gradients that are not caused by deformation are reported by Maeder *et al.* [60], who performed laser annealing of silicon. According to a theory by Raabe *et al.* [64]

plastically strained crystals can develop intra-granular orientation gradients. Thus most of the GND analysis in literature is concerned with mechanical properties and the effect of deformation on mechanical properties, and most reported orientation gradients occur as a result of deformation.

2.3 Genetic Algorithms

The standard by which artificial intelligence programs are judged is known as the Turing Test, originally called The Imitation Game, which Turing proposed in his seminal paper “Computing Machinery and Intelligence” [65]. In the same paper he also proposed the idea of “learning machines” that mimic the process of evolution, which gave rise to the field of evolutionary computation [66]; the history of evolutionary computation has been detailed by Fogel [67]. Genetic algorithms are widely considered [68] to have begun with Holland’s “Adaptation in Natural and Artificial Systems” [69], and are now an established subfield of evolutionary computation [66]. Genetic algorithms have been successful in an exceptional number of fields [70–75], including materials science [76–78].

Genetic algorithms (GAs) are an optimization technique that mimics evolution by incorporating the principles of natural selection, inheritance of genetic material, crossover, and mutation. Candidate solutions, called individuals, pass down their genetic material, represented as parameter encodings, to a number of offspring based on how fit each individual is. Each individual is a potential solution generated by the algorithm, and all of the individuals are collectively referred to as a population. Each iteration of the algorithm produces a new population, and the population from a particular iteration of the algorithm is called a generation. Each individual is evaluated according to a fitness function that determines how fit that individual is. Examples of criteria that may be used in a fitness function include cost, weight, and efficiency. It is possible to evaluate fitness based on multiple criteria, and an algorithm that does this is called a multi-objective genetic algorithm [79]. An individual with

a high fitness will produce more offspring than one with a low fitness, and, as in evolution, this ensures that beneficial genes are more likely to be passed down from one generation to the next and spread throughout the population. The process by which individuals produce offspring is called crossover, which combines the genetic material from two individuals. Another feature of GAs is mutation, which makes small, random changes to the encoding of an individual. This introduces new genetic material to the population, which helps prevent premature convergence of the algorithm.

GAs have several advantages over other optimization techniques. In particular: they are less likely than other techniques to converge prematurely; they can work well with an unintuitive and complex solution space; they are inherently parallelizable; and they tend to produce robust solutions. They are less likely to converge prematurely because, if implemented properly, they start with and maintain a diverse population, which allows them to explore a larger area of the search space. GAs make no restrictive assumptions about the search space of any particular problem, which allows for success in a diverse range of fields [68]. This makes them useful for early attempts at solving a problem when little is known because all assumptions come from the programmer and not from the basic structure of the algorithm. Unlike many optimization techniques—even similar ones such as simulated annealing—GAs are easy to parallelize, because the most computationally intensive section of the algorithm (the fitness function) can be evaluated separately for each individual. Evaluating each individual can be done without any dependence on other individuals or previous generations, unlike in other optimization techniques where solutions may be interdependent. Lastly, GAs often produce robust solutions in the sense that small changes to the encoding of individuals after many generations will not typically have a drastic effect on its fitness. This is because once the population converges after hundreds of generations, for any small, single change in the parameter encodings of an individual, it is probable that a similar change was previously made, and if such a change substantially lowered the fitness of that individual then it would have been unlikely to produce offspring. This is not always true, however: a

counterexample is binary encoding, where any changes made to an individuals encoding are inherently discretized and cannot be made arbitrarily small.

The problem of dislocation structure generation satisfies all of the informal criteria typically used to determine if a GA is the best approach for a particular problem. These criteria include a complex, nonintuitive solution space with variables that interact in unknown ways; a good way to encode and evaluate potential solutions; and a need for a sufficiently good solution but not the absolute best. The solution space of this problem, which will be discussed in more detail later, is complex and unintuitive because it is difficult to determine the effect that changing the shape and Burgers vector of a dislocation will have on the resulting orientation map without performing the computation. Optimizing these orientation maps against dislocation density is also a difficult problem. For this problem it is easy to construct a good fitness function, such as a combination of the average misorientation between the orientation map produced by a particular set of dislocations and that of the smooth model from the previous section and the total dislocation density. It is also relatively straightforward to encode potential solutions; for example, we may store each dislocation as an array of (x, y) coordinates and its Burgers vector as an array with three elements. It is not likely that this problem has a unique solution, *i.e.* it is probable that there are multiple different dislocation configurations that yield indistinguishably good orientation maps and that have comparable dislocation densities; thus it is not necessary to find the absolute best solution. Additionally, it would not necessarily be clear if one of these indistinguishably good solutions were found because we do not have a means of verifying that a particular solution cannot be improved.

Chapter 3:

Methods

3.1 Plotting Orientation Maps

The orientation data provided by an EBSD machine typically come in one of two forms (or sometimes both), an orientation matrix or Euler angles. An $n \times n$ orientation matrix (for n dimensions, almost always 3) is also called a direction cosine matrix, and describes a relation between two sets of axes. Typically these sets of axes are the crystal axes and the samples axes. However all 9 elements of this matrix are not all independent, as there are only three degrees of freedom for a rotation. This leads to an alternate scheme for representing rotations, known as Euler angles. Euler angles are a triplet specifying rotations about a particular set of axes. Often the first and third axes will be the same; one scheme is to first rotate about the z axis, then about the x axis, and then about the rotated z axis. This scheme is known as Bunge, and is the most common. It is possible to convert between a rotation matrix and Euler angles, and some software packages designed for scientists or engineers include functions or methods for this conversion. It is important to know which scheme is being used when converting, however, as using the wrong scheme may produce nonsensical results. Orientation matrices are popular for two reasons. The first is that multiplying orientation matrices is the same as concatenating rotations (*i.e.*, $[R_2][R_1][M]$ is equivalent to rotating the orientation represented by $[M]$ first by rotation $[R_1]$ and then $[R_2]$; it is important to remember that matrix multiplication is not commutative). The second reason is that matrix multiplication is a very common operation, and so most software (and some hardware) is optimized for it, which makes these operations very fast. An interesting property that arises from rotation matrices is that orientations and rotations are equivalent, as any orientation can be described as a rotation from the reference axes (in other words,

any rotation matrix is equivalent to itself times the identity matrix).

In addition to the orientation data, the EBSD machine will output the x and y coordinates at each point, and sometimes other data that are not relevant to displaying the orientation. A common way to view orientation data is to take the out-of-plane (OOP) component of the orientation and assign it a color under some color scheme. The OOP component can be thought of as either the lattice planes lying in the plane of the film or as the direction perpendicular to the film. A useful property of orientation matrices is that the OOP component of the orientation is just the last row of the matrix. The orientation matrix has no entries larger than 1, so to convert the component to Miller indices simply scale the values until they are integers. For example, suppose the OOP component is $\left[\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right] \approx [0.577, 0.577, 0.577]$. To convert this to a Miller index, simply scale by $\sqrt{3}$, which gives $[1\ 1\ 1]$. The most common way to color orientations is with the stereographic triangle, which can be seen in Figure 2.1. In the stereographic triangle, $\{001\}$ planes (or $\langle 001 \rangle$ directions) are colored red, $\{011\}$ is green, and $\{111\}$ is blue. The basic procedure for converting Miller indices to a color, with v as the 3-element vector of Miller indices, is to sort the absolute value of v as $v_s = \text{sort}(\text{abs}(v))$, then create a new vector c such that $c(1) = v_s(3) - v_s(2)$, $c(2) = v_s(2) - v_s(1)$, $c(3) = v_s(1)$, and then normalize c so its maximum component is 1 (*i.e.*, if $\max(c) \neq 0$, $c = c/\max(c)$). If the maximum component is 0, then the entire vector is zeros and it has no orientation. This yields c with red, green, and blue components in order, on a scale of 0 to 1, though some programs may require that color values be an integer between 0 and 255.

Another common way to view orientation data is with pole figures. Pole figures are more commonly used to analyze three-dimensional orientation data, and there are readily available software packages capable of plotting pole figures. See [80] for more information about pole figures. An excellent tool for plotting pole figures, as well as general texture analysis, is MTEX,* [81] which is a Matlab toolbox. MTEX is capable of plotting pole figures, inverse

* available at <https://github.com/mtex-toolbox/mtex>

pole figures, orientation distribution functions, and much more. MTEX was used to make the pole figures in this thesis.

3.2 Geometrically Necessary Dislocations

The Nye tensor, sometimes called a dislocation tensor or a dislocation density tensor, represents a continuum approach to crystal plasticity and is a second-order tensor that relates the closure failure of a Burgers circuit with the area it encloses as $B_i = \alpha_{ij} n_j dA$ [35], where B is the closure failure of the Burgers circuit (the Burgers vector), α is the Nye tensor, n is the unit normal vector to the Burgers circuit, and dA is the area enclosed by it. A more useful representation of it is $\alpha = \kappa^\top - \frac{1}{2} \text{trace}(\kappa)$, where κ is the curvature tensor. The curvature tensor is defined as $\kappa_{ij} = \frac{\partial \phi_i}{\partial x_j}$, usually approximated as $\frac{\Delta \phi_i}{\Delta x_j}$, where $\Delta \phi_i$ is the infinitesimal rotation about axis i and Δx_j is the change in position along axis j . The diagonal components of the curvature tensor correspond to twisting of the lattice, while the off-diagonal components correspond to bending. The method that we used, below, is based on that of Kysar *et. al* [42].

The misorientation between adjacent regions of the film (pixels) is calculated as $[\Delta g_{AB}] = [g_B] [g_A]^{-1}$, where $[\Delta g_{AB}]$ is the misorientation matrix and $[g_B]$ & $[g_A]$ are the respective orientation matrices for the previous and current pixels. This misorientation is then converted to axis-angle form, which is calculated as

$$n_1 = \frac{[\Delta g_{AB}]_{23} - [\Delta g_{AB}]_{32}}{\text{norm}}, \quad n_2 = \frac{[\Delta g_{AB}]_{31} - [\Delta g_{AB}]_{13}}{\text{norm}}, \quad n_3 = \frac{[\Delta g_{AB}]_{12} - [\Delta g_{AB}]_{21}}{\text{norm}}$$

$$\text{norm} = \sqrt{([\Delta g_{AB}]_{23} - [\Delta g_{AB}]_{32})^2 + ([\Delta g_{AB}]_{31} - [\Delta g_{AB}]_{13})^2 + ([\Delta g_{AB}]_{12} - [\Delta g_{AB}]_{21})^2}$$

$$\omega = \arccos \left(\frac{1}{2} (\text{trace}([\Delta g_{AB}]) - 1) \right)$$

where $[\Delta g_{AB}]$ is the misorientation matrix, n is the axis of misorientation, and ω is the angle

of misorientation. The axis-angle pairs then form the components of the curvature tensor for each pixel such that the first column of the curvature tensor is $n_x\omega_x$ and the second column is $n_y\omega_y$, since misorientation is calculated both horizontally and vertically.[†] The curvature tensor is in turn used to calculate the Nye tensor for each pixel as $\alpha_{ij} = \kappa_{ji} - \text{trace}(\kappa)$. The result is then expressed in crystal (local) coordinates: $[\alpha'] = [g_A][\alpha][g_A]^\top$. The Nye tensor $[\alpha']$ is related to GND density by $[\alpha'] = [A][\rho]$, where $[A]$ is a coefficient matrix and $[\rho]$ is the GND density [42].

In order to calculate $[A]$, the Burgers and sense vectors for each slip system must be determined, calculated, and normalized. α tantalum has a BCC crystal structure, so $\{110\}\langle 111\rangle$ -type directions are the primary slip systems.[‡] The Burgers vectors lie along the slip directions and the sense vectors are calculated as the cross product of the normalized indices of the plane of the slip system and the normalized Burgers vector. Once the Burgers and sense vectors are known, the coefficient matrix $[A]$ can be calculated such that $[A]_{ij}^k = a \vec{b}_i^k \vec{\xi}_j^k$, where k is the slip system, a is the unit cell length, and \vec{b} & $\vec{\xi}$ are the respective Burgers and sense (tangent) vectors. Once $[A]$ is calculated, $[\alpha'] = [A][\rho]$ can be rearranged as $[\rho] = \text{Pinv}[A][\alpha']$, where $\text{Pinv}[A]$ is the Moore-Penrose pseudoinverse of $[A]$. This yields the GND density for each pixel for each slip system. Finally, the GND density at each pixel is summed across all slip systems to yield the total GND density.

3.3 The Smooth Model

There are three indications that the phase transformation is a radial process. The first can be seen in Figure 2.1e that there are several locations where low- and high-angle orientation boundaries appear to extend outward radially from a single point. In order to examine this in more detail we plotted the orientation at each point and we obtained Figure 3.1. This

[†] There is no third column and thus no depth because the data come from a two-dimensional film surface

[‡] While the $\{123\}\langle 111\rangle$ & $\{112\}\langle 111\rangle$ slip systems are very close in energy to the $\{110\}\langle 111\rangle$ slip system in BCC crystals, it has been shown that for pure tantalum annealed in UHV the $\{110\}\langle 111\rangle$ systems dominate [82]

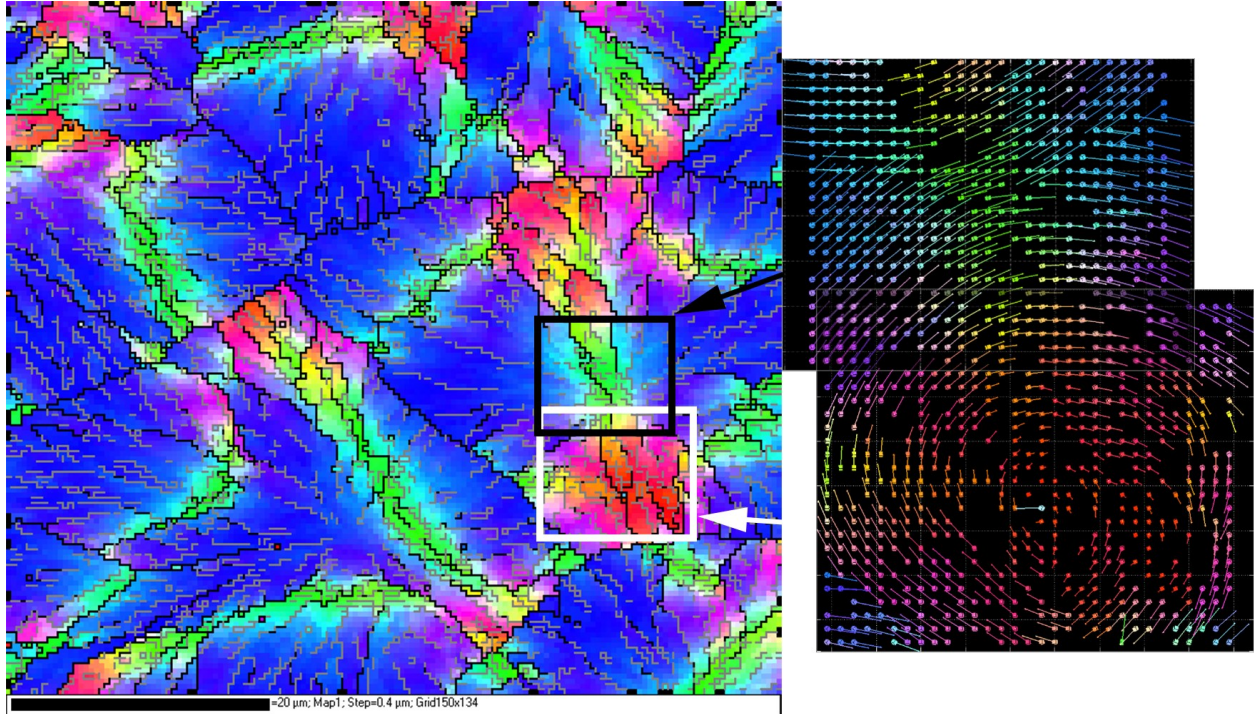


Figure 3.1: Orientation data from the indicated regions of the film in Figure 2.1e. At each point the orientation is plotted with the OOP color and a vector pointing in the (001) direction in crystal (local) coordinates. This unusual pattern is an indication of a radial transformation process.

figure shows a plot of the orientation at each point in the indicated regions of the film in Figure 2.1e with the color corresponding to the OOP component and a vector plotted in the (001) direction in local (crystal) coordinates. The unusual circular pattern that forms is the second indication that the phase transformation is a radial process. One way this pattern could form is if the orientation rotates outward radially from the center about a rotation axis perpendicular to the radial direction. The third indication that the phase transformation is radial is a repeating triangular pattern that can be seen in the film. This pattern is has a (111) (blue) center, (011) (green) sides, and (001) (red) corners, and if a cube were to be colored accordingly (*i.e.* red faces, green edges, and blue corners) and viewed along a body diagonal (*i.e.* looking at a corner) it has the same symmetry.

We did these computations and obtained Figure 3.2. These calculations used a constant rotation rate, meaning that the amount of rotation at each point was linearly dependent on its distance from the origin. The basic procedure for this is as follows:

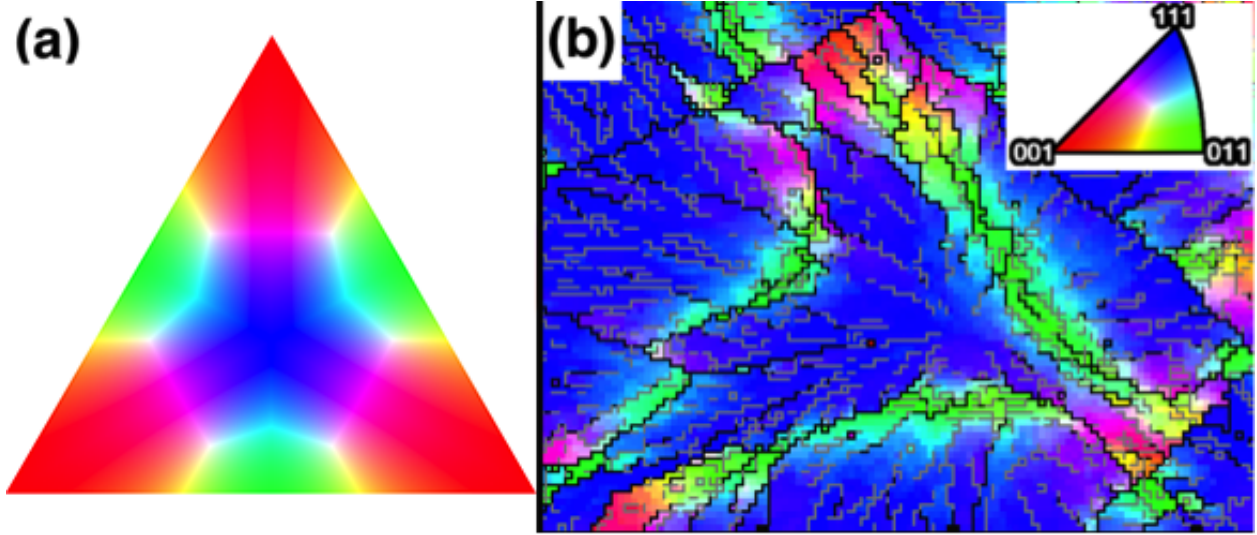


Figure 3.2: A simple radial rotation model, which starts in the center and rotates outward radially, rotating about an axis perpendicular to the radial direction. Both are colored by their OOP orientation component according to the inset stereographic triangle. (a) is the result from the simple model and (b) is the corresponding region of the film data. The match of symmetry is an indication that the phase transformation is a radial process.

1. Choose a starting orientation matrix $[s]$ for the center pixel.
2. Select the desired pixel at which to compute the orientation.
3. Compute the rotation axis \vec{v}_r as the in-plane vector perpendicular to the radial vector \vec{r} (the vector from the center pixel to the selected point), $\vec{v}_r = |\text{vec}x_3 \times \vec{r}|$ where \vec{x}_3 is the vector perpendicular to the plane of the film.
4. Scale the distance from the center pixel $||\vec{r}||$ by a constant factor c to get the magnitude of the rotation ω , $\omega = c||\vec{r}||$.
5. Convert the axis-angle form of the rotation to a rotation matrix, for example using the `vrrotvec2mat` function in Matlab, $[R] = \text{vrrotvec2mat}(\vec{v}_r, \omega)$.
6. Rotate the starting orientation matrix $[s]$ about the rotation axis \vec{v}_r by the computed amount ω using the computed rotation matrix $[R]$ to get the orientation at the selected point, $[o] = [R][s]$.
7. Repeat steps 2–6 for all desired pixels.

While this simple model matches the symmetry of the film, it does not match the spatial distribution of the different orientations (*i.e.* the (1 1 1) center needs to be larger, the (0 1 1) edges need to be longer, and the (0 0 1) corners need to be smaller). In order to accomplish this, we modified the model to use a variable rotation rate. The variable-rotation-rate model has two components, the angular component and the radial component. The angular component is the component of the model that modifies the rotation axes and the radial

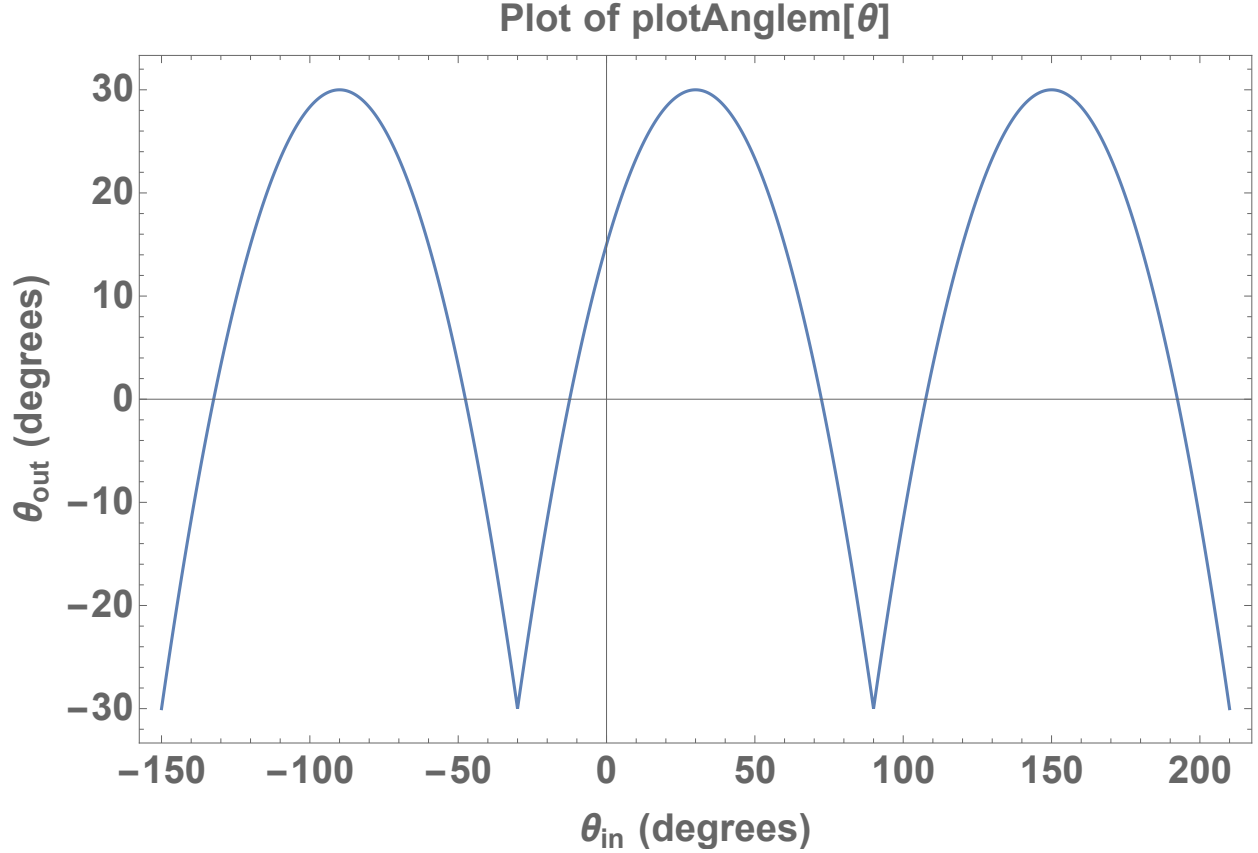


Figure 3.3: A plot of the Equation (3.1), which broadens the (0 1 1) stripes and shrinks the (0 0 1) regions. Note that the profile is broader near -90° , 30° , and 150° , which correspond to (0 1 1) stripes on the triangle, and that it is narrower near -30° , 90° , and 210° , which correspond with the (0 0 1) regions at the corners of the triangle.

component is the component that modifies the amount of rotation. The angular component is necessary to widen the (0 1 1) regions into stripes and to shrink the (0 0 1) regions into corners. The radial component is necessary in order to expand the (1 1 1) region so that the relative areas of the (111), (0 1 1) and (0 0 1) regions are similar to those in the film. The angular component of the model is

$$\theta_{\text{out}} = 15 \cos[\theta_{\text{in}}/19] + \theta_{\text{in}} \quad (3.1)$$

where both θ_{in} and θ_{out} are in degrees. θ_{in} is the angle that the radial vector (step 3 above) makes with the horizontal, and θ_{out} is the angle that is actually used to compute the rotation axis. A plot of this function is shown in Figure 3.3.

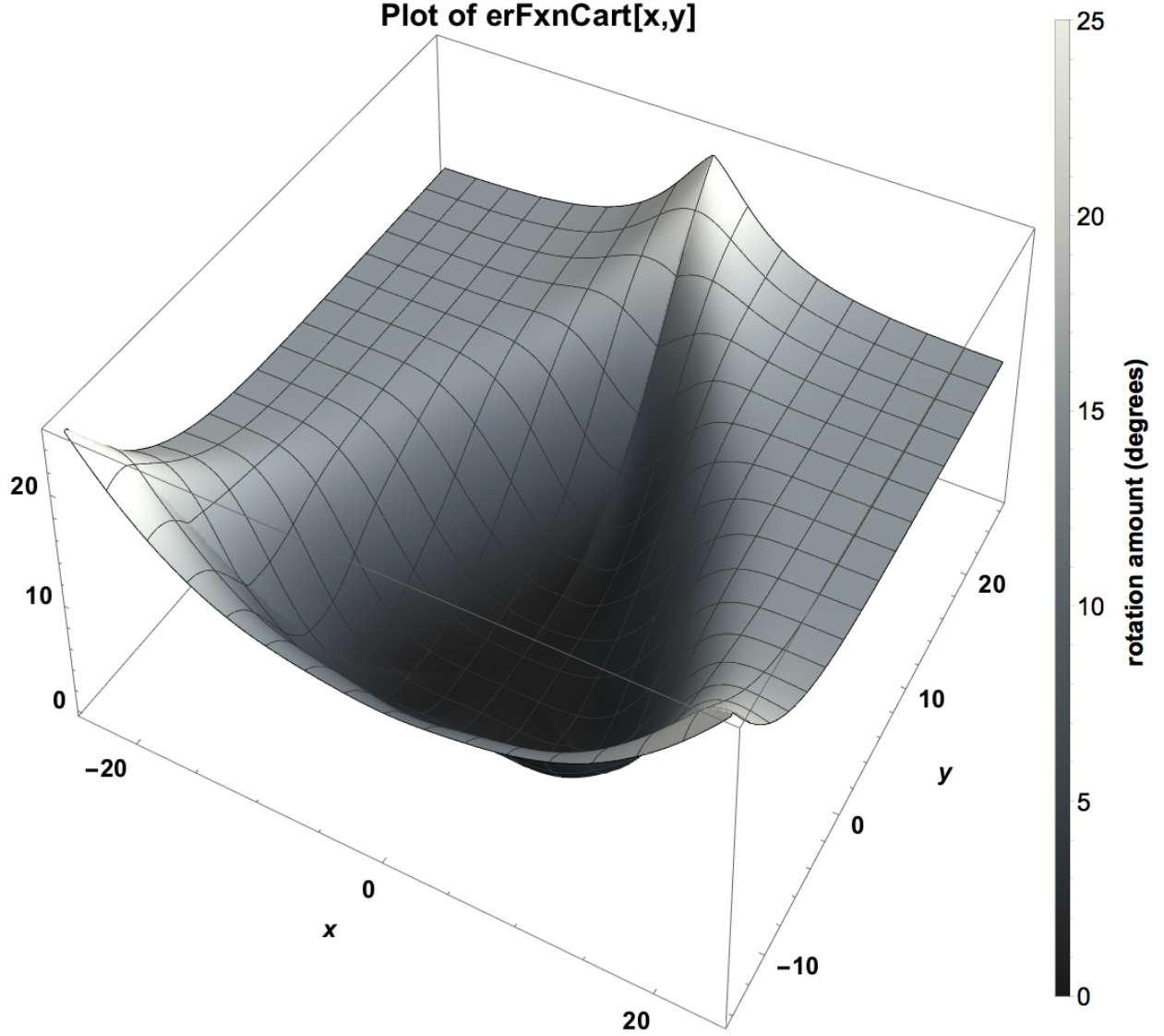


Figure 3.4: A plot of Equation (3.2). The value of this function gives the magnitude of rotation at each point. Note the threefold symmetry and the sigmoidal profile along any path extending radially from the origin.

The other component, the radial component, is significantly more complicated. Its desired effect is to increase the area of the (111) region, which means that it must be low in the center of the triangular region for a moderate area, then smoothly and sharply increase. It must sharply increase because, in the film, transitions from (111) to either (011) or (001) are relatively short; the amount of (221) and (211) (cyan and magenta, respectively, on the EBSD color mapping) is small, corresponding to a high rotation rate at those points. In order to achieve this smooth yet rapid rise, rotation along any radial line was modeled using

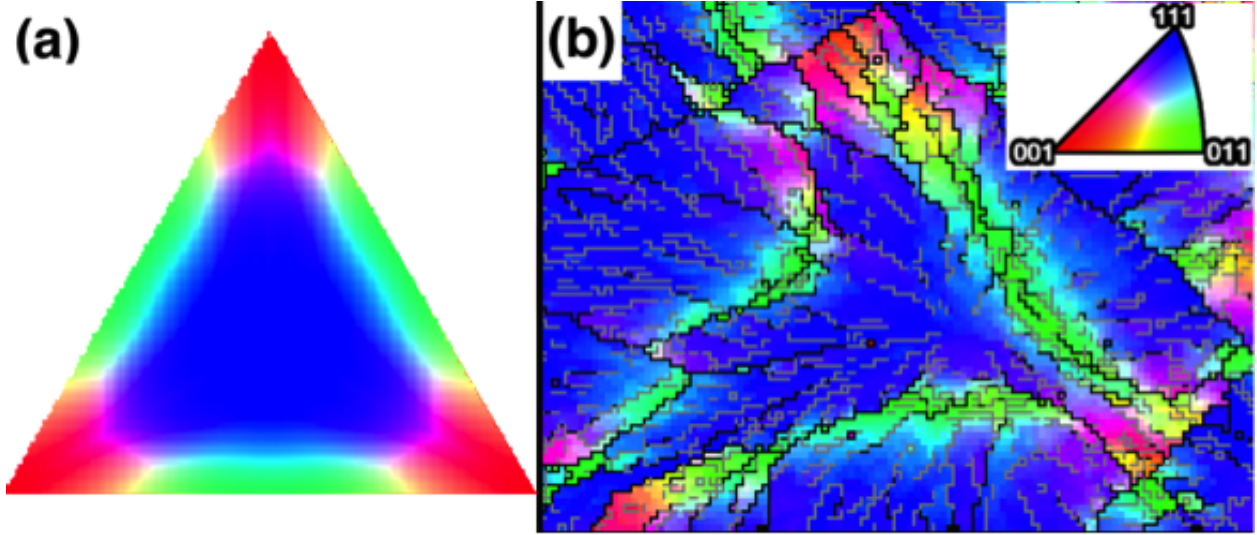


Figure 3.5: A radial rotation model with the adjustments incorporated. The model data (a) now not only match the symmetry of the film data (b), but also the spatial distribution of orientations. The close match orientations is an indication that the phase transformation is a radial process.

an error function (resulting in a sigmoidal profile). The magnitude of the final rotation at any angle is determined by the ending orientation on the triangle at that angle. For example, the path along radial vector at 30° must end at the $(0\ 1\ 1)$ orientation, corresponding to a total rotation of $\arccos\left[\sqrt{2/3}\right] \approx 35.264^\circ$. Similarly, the path along radial vector at 90° must end at $(0\ 0\ 1)$, corresponding to a rotation of $\arccos\left[\sqrt{1/3}\right] \approx 54.736^\circ$. This is then used to scale the magnitude of the sigmoidal function, yielding

$$\omega = \frac{d}{\sqrt{2}} \sqrt{\left(1 + \operatorname{erf}\left[\frac{\left(r - 5 - 4 * (\cos[30 * \text{TriangleWave}[\theta_{\text{out}}/240 + 1/8]])^{40}\right)}{(d + 10^{-20})/4} - 2\right]\right)} \quad (3.2)$$

where ω is the amount to rotate at some point (x, y) , r is the distance from that point to the center $(r = \sqrt{x^2 + y^2})$, θ_{out} is from the previous equation, $d = \operatorname{ArcTan}[\sqrt{2}]/2 (\sqrt{3} \sin[2\theta_{\text{out}}] + 2 \cos[2\theta_{\text{out}}])^{-1/2}$, and $\text{TriangleWave}[\theta]$ is a function that produces a wave of straight lines that vary between -1 and 1 with the same period as $\sin[\theta]$. As with the previous equation, all angles are in degrees. This is an empirically derived formula, and as such there is likely a significantly simpler form that would behave similarly. However, this formula meets the initial and boundary conditions, and as such is sufficient for this work. A plot of this function is shown in Figure 3.4.

With these adjustments, the model now produces the result in Figure 3.5. The model now matches not only the symmetry of the film data but also the spatial distribution of the different orientations.

3.4 The Genetic Algorithm Model

Attempting to determine directly from the orientation data what dislocations are present has proven to be infeasible for several reasons, so the opposite approach of generating dislocation structures, seeing how well they match the smooth model data, and then improving them is a good approach to take. For this type of iterative approach a genetic algorithm works very well. A genetic algorithm mimics the process of evolution, but ‘evolving’ solutions to a problem instead of organisms. The basic terminology of a genetic algorithm is fairly intuitive:

individual: a candidate solution to the problem (in this algorithm a set of dislocations)

population: all of the individuals collectively

generation: an iteration of the algorithm

fitness: a quantitative measure of how good each individual is (often includes cost, weight, or efficiency; in this algorithm misorientation and dislocation density)

genetic material: the parameter values of an individual (in this algorithm, the location and shapes of the dislocations and their Burgers vectors)

crossover: the process by which individuals exchange genetic material to produce new offspring; the number of offspring each individual has is proportional to its fitness (the principle of natural selection)

mutation: occasional, small, random changes made to the genetic material of an individual; serves to add new genetic material to the population on which natural selection can act in order to prevent premature convergence

convergence: once the fitness of the population stops improving, it is said to have converged; typically all of the individuals will be very similar by this point. If this happens too early then the solution will not be good enough. The population may not always converge, for example if computing resources are exhausted

The basic outline of the procedure that a genetic algorithm uses is as follows:

1. **Initialization** The first step requires the creation of a population for the first generation. Typically all of the parameters will be randomly generated unless there is some knowledge or intuition about what values certain parameters should or should not have.
2. **Fitness Evaluation** The fitness of every individual in the population must be determined by the fitness function. This is the most computationally intensive component of the algorithm (except for trivial examples such as the onemax problem[§]).
3. **Reproduction** A new generation must be created, with the same population size as the previous generation. This is done by selecting individuals pairwise and exchanging some genetic material between them. The number of times this happens for a particular individual depends on its fitness.
4. **Mutation** The algorithm will randomly decide to make very small changes to the encoding of some individuals.
5. Repeat steps 2–4 until the population converges or some other stopping condition is met (*e.g.* sufficiently good fitness is achieved or compute time is reached).

The following list contains the basic outline of the important components of the genetic algorithm, with 1. as the population initialization, 2. as the fitness evaluation function, and 3. as tournament selection. These correspond to steps 1–3 in the list above.

1. **Initialization** Create the population by randomly assigning each of the following parameters for each individual:

[§] https://deap.readthedocs.io/en/master/examples/ga_onemax.html

- (a) location of gaussian noise added to level set function
- (b) the location of each of the OOP dislocations
- (c) the Burgers vectors of each OOP dislocation (all $\langle 111 \rangle$ type)
- (d) the number of IP dislocations (contour levels)
- (e) the Burgers vector of each of the IP dislocations (all $\langle 111 \rangle$ type)

2. **Fitness Evaluation** Evaluate the fitness of each individual. For this algorithm it is a combination of the misorientation between the smooth model data and the orientation produced by the set of dislocations of each individual.

- (a) add 2D gaussian noise to landscape at specified points
- (b) take evenly spaced contours at specified number of levels, these become IP dislocations
- (c) assign each IP dislocation its Burgers vector from the individual's list
- (d) evaluate orientation at each point (loop over θ and then r):
 - i. if $\mathbf{r}==0$: assign start orientation (at center of triangle)
 else: take the radial line from the previous point and find all intersections with the IP dislocations
 - ii. check if there are any OOP dislocations at the point
 - iii. set α to a 3×3 zero-matrix, and for each crossed dislocation:
 - A. get $\vec{\xi}$ as $[001]$ for OOP dislocation or the tangent vector to the IP dislocation at the point of intersection
 - B. rotate \vec{b} into global coordinates (multiply by orientation matrix of previous point) and multiply by length of Burgers vector in Ta
 - C. $\rho = L/(A * t_f)$ (L is the length of the dislocation line, A is the area between it and the next dislocation, and t_f is the film thickness)

$$\text{D. } \alpha_{ij} += \rho \vec{b}_i \vec{\xi}_j$$

- iv. $\kappa = \alpha^\top - 1/2 \text{trace}(\alpha)$
- v. $[\omega] = [\kappa] * dr$ (dr is vector from previous point)
- vi. $\omega = ||[\omega]||, n = [\omega]/\omega$ (axis-angle)
- vii. convert axis-angle to rotation matrix and multiply by previous point's orientation matrix to get current orientation
- viii. get misorientation matrix between current orientation and smooth model at same point
- ix. convert misorientation matrix to axis-angle form and store the misorientation angle for this point
- (e) compute dislocation density (total line length of all IP and OOP dislocations, including bundle size, divided by triangle volume)
- (f) compute average misorientation across all points in the triangle
- (g) merge the above into the fitness according to their relative weights

3. Reproduction and Mutation Create new offspring by exchanging genetic material between individuals pairwise. For this algorithm tournament selection is used to determine the number of offspring each individual has. Tournament selection repeatedly selects k individuals and keeps only the most fit from each selection, until it has as many individuals as the size of the population it started with. It then crosses every other individual with its neighbor. Mutation is applied afterwards, and this is all repeated.

- (a) select k individuals n times (n is desired population size, k controls selection pressure), keeping only the most fit individual from each selection
- (b) pair up every other offspring with its neighbor (1 with 2, 3 with 4, etc...)

- (c) for each pair of offspring, if `rand()` > `cxbp`, crossover the two in place (`cxbp` is the probability of crossover)
- (d) for each offspring, if `rand()` > `mutpb`, mutate the individual (`mutpb` is the probability of mutation)
- (e) offspring becomes new population

Chapter 4:

A Genetic Algorithm for Dislocation Structure Prediction in Phase-Transformed Tantalum Thin Films

Ari Kestenbaum,^{*} Elizabeth Ellis,[†] and Shefford Baker^{*†}

Abstract

Tantalum thin films are used in a wide range of applications, from micro- and nanoelectronics to military and aerospace applications, to medical devices. Tantalum has two phases, the stable bulk α phase and the metastable β phase found only in thin films. In a recent discovery, it has been shown that when tantalum is phase-transformed from β to α , it can form a unique microstructure with continuous changes in orientation and discontinuous grain boundaries. Dislocation structures must be present in order to account for the lattice curvature associated with orientation gradients. To identify the dislocation arrays that give rise to this microstructure, we first analyzed the electron backscatter diffraction (EBSD) data of the orientation gradients using geometrically necessary dislocations (GNDs), but we found that our EBSD data are too noisy and do not have sufficient resolution for this approach. In this work, two models are presented in an attempt to describe the dislocation structures that must be present to account for the orientation gradients. In the first, a method of generating smooth, noise-free orientation data that match key features of the actual phase-transformed microstructure is developed and validated by comparing model data with film data. In the second model, a genetic algorithm approach is used to generate dislocation structures that can account for the orientation gradients produced using the

^{*} Cornell University, Department of Materials Science and Engineering, Bard Hall, Ithaca, NY 14853

[†] Cornell University, Department of Theoretical and Applied Mechanics, Upson Hall, Ithaca, NY 14853

first model. The genetic algorithm model consistently generates dislocation structures that produce orientation maps with an average misorientation of less than 2° from the smooth orientation gradients generated by the first model. The generated dislocation structure is consistent across multiple runs, with similar shapes and Burgers vectors. The genetic algorithm model selects two types of dislocations, mostly ($\sim 80\%$) $[\bar{1}\bar{1}1]$ and the rest $[\bar{1}11]$. The dominance of these two slip systems provides a starting point to search for a mechanism in the β -to- α phase transformation that is capable of producing these dislocations.

4.1 Introduction

Tantalum thin films are used in a wide variety of applications due to their useful properties. In a thin film, tantalum can be deposited in one of two phases, α or β . The two have very different properties, and thus are suited to different applications. The stable bulk α phase is ductile, has a low resistivity, and has a BCC crystal structure [7]. α -Ta films are used as a diffusion barrier between copper and silicon in microelectronics [8–11], in thin-film capacitors [12, 13], and in high-temperature wear-resistant coatings [83]. The β phase, by contrast, has a self-hosting σ -type Frank-Kasper structure [16], is only found in thin films, is brittle, has a high resistivity [7], and has a low temperature coefficient of resistance (TCR) [13]. It is often used in thin-film resistors [12, 13, 15] because of its high resistivity [7] and low TCR [13]. It is also susceptible to reactive ion etching [19] and, because tantalum absorbs x-rays well, β tantalum is used in x-ray lithography masks [19, 20]. Differences between the phases are summarized in Table 4.1. β tantalum is also metastable [7, 21] and will transform to α when heated [24, 27].

The phase transformation from β to α tantalum has been reported to begin at temperatures as low as 300°C [23, 24] and may not finish until above 800°C [25, 27]. Knepper *et al.* showed that this dramatic temperature variation occurs because the transformation process is very sensitive to oxygen in the film [24]. Although it has been proposed that the phase

Table 4.1: Comparison of properties of α and β tantalum. Data from [7] except where noted.

Phase	α -Ta	β -Ta
Structure	BCC $Im\bar{3}m$	Tetragonal $P\bar{4}2_1m$
Lattice	a=b=c=	a=b=1.0211
Parameters (nm)	0.33058	c=0.53064
Ductility	Ductile	Brittle
Hardness (KHN)	200–400	1000–1300
Resistivity ($\mu\Omega\cdot\text{cm}$)	15–60	170–210
TCR (10^{-6} K^{-1}) [13]	1100–3400	-150 ± 30

transformation might be martensitic [27], a shear transformation is insufficient to account for a σ -to-BCC phase transformation [84] and cannot account for the large change in grain sizes observed [32]. It was further shown by Baker *et al.* [32] that the phase transformation mechanism must be nucleation and growth, as the process is thermally activated and kinetically limited, and highly temperature dependent. There is also a large stress change associated with the phase transformation [1, 24, 27–32], so this phase transformation, whether desired or not, can have a large effect on device performance. Thus it is important to understand this phase transformation from both a scientific and an industrial perspective.

The microstructure that arises as a result of this phase transformation is unique [1, 32]. Figure 4.1 shows electron backscatter diffraction (EBSD) orientation data, colored by out-of-plane (OOP) orientation according to the inset stereographic triangle, for a series of phase-transformed tantalum films 400–500 nm thick, sputtered at argon pressures of 14, 15, and 16 mTorr for Figures 4.1a, 4.1b, and 4.1c respectively [1]. These samples are pure tantalum and were deposited and annealed in ultra-high vacuum (UHV) [1]. This microstructure exhibits continuous orientation gradients that can extend up to tens of μm and transition from (001) to (111) without crossing any low- or high-angle boundaries, a change of more than 50° .

High-angle boundaries are shown in Figure 4.1 as black lines, and denote a change in orientation of more than 8° across the boundary; low-angle boundaries are gray lines and denote a change of between 4 and 8° . These boundaries are discontinuous, which makes it difficult to define a grain in the traditional sense, so the term “pseudo-grain” will be used to refer to a region of the film that is completely surrounded by a high-angle boundary.

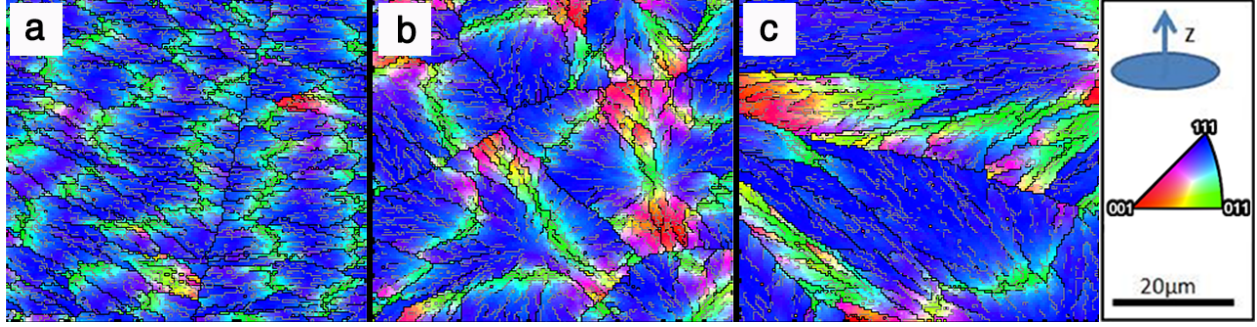


Figure 4.1: Out-of-plane (OOP) orientation of a series of phase-transformed tantalum films, sputtered at argon pressures of 14, 15, and 16 mTorr for (a), (b), and (c) respectively [1]. Colors correspond to orientation given by the inset stereographic color triangle; black and gray lines denote high- and low-angle boundaries. Distinct features include continuous orientation gradients and discontinuous grain boundaries.

Additionally, high- and low-angle boundaries will be referred to as orientation boundaries instead of grain boundaries. The discontinuous orientation boundaries sometimes start in the middle of a pseudo-grain as a low-angle boundary, transition to a high-angle boundary, then become a low-angle boundary again, and ultimately end in the middle of a different pseudo-grain. This microstructure is completely unlike the canonical columnar thin-film microstructure in which grains of a single orientation are separated by continuous high-angle boundaries [34]. This microstructure and the experimental parameters for these films are described in more detail by Baker *et al.* [1, 32].

In order to better understand this phase transformation, it is first necessary to understand the associated dislocation structure. We first used a Nye tensor analysis on the EBSD data in an attempt to compute the density of geometrically necessary dislocations (GNDs); but as described in Section 4.2.2 we found that our data are too noisy and do not have sufficient spatial resolution. As a next step, we then developed a model microstructure that closely matches the observed microstructure but has mathematically smooth and continuous changes in orientation. We analyzed the smooth model data using GNDs but found that, as discussed in Section 4.3.2, it does not yield useful information because the rotations in the model are not associated with specific physical dislocations. Using this smooth orientation data we then took the opposite approach; instead of attempting to determine directly from the orientation

data what dislocations could account for it, we developed a model that produces dislocation structures, computes how well the resulting orientation data match the smooth model data, and then modifies the dislocation structure to better match the smooth model data, using a genetic algorithm.

Genetic algorithms (GAs) are descended from Alan Turings “learning machine” which parallels the principles of evolution and was a core component in his “imitation game”, which is now known as the Turing test [65]. GAs are a subset of evolutionary computation, the early history of which has been chronicled by Fogel [67] and Goldberg [68]. Hollands *Adaptation in Natural and Artificial Systems* [69] is the primary monograph on the topic, and is widely considered to have initiated the field of study [68]. GAs have been successful in a remarkable number of fields, including the design and optimization of race cars [70], satellite antennas [71], molecular geometry [72], thermal control of buildings [73], water distribution networks [74], and electromagnetic devices [75]. Materials science applications include generating grain boundary configurations in bicrystals [76], interface structure prediction in multicomponent systems [77], and finding low-energy structures of symmetric tilted grain boundaries in Si [78]. However, no GA has previously been created for dislocation structure generation.

GAs have a number of advantages over other optimization techniques: they perform better than other techniques when the optimization criteria are complex or their interplay is unknown, they can be better at approaching the global optimum and are less likely to converge prematurely because they start with a diverse initial population, and they do not require a smooth or even continuous function to optimize, which is useful when computing discrete entities, such as dislocations.

The remainder of this paper is structured as follows: an overview of GNDs, a description of the Nye tensor GND density computation, and its results; motivation for the smooth model, a description of it, and its results; motivation for the genetic algorithm model, a description of it, and its results; a discussion of the results from the genetic algorithm model; and conclusions and ideas for future work.

4.2 Geometrically Necessary Dislocations

Dislocations that are required for lattice curvature are known as geometrically necessary dislocations (GNDs) and were first formulated as a tensor describing the state of dislocation at a point by Nye [35]. The concept was extended by Kröner [36,37] and Ashby [38], with important contributions to the GND framework by Arsenlis and Parks [39], Sun *et al.* [40], El-Dasher *et al.* [41], and Kysar *et al.* [42]. The GND framework has been applied to both bulk materials [40–45] and thin films [46–49] as a way to understand strain gradient plasticity [50–53]. Orientation gradients have been observed in many bulk samples [54–59] and some thin films [60–63], and are sometimes reported with GND analysis [57–60] and sometimes not [55–57,61–63]. GND analysis performed on samples with orientation gradients is generally concerned with the effect of GNDs on hardness and yield stress, as nearly all of these samples have been heavily deformed or highly strained. The only previously observed orientation gradients that are not caused by deformation are reported by Maeder *et al.* [60], who performed laser annealing of silicon. Nearly all observed orientation gradients are observed in heavily deformed samples because according to a theory by Raabe *et al.* [64] plastically strained crystals can develop intra-granular orientation gradients. Thus most of the GND analysis in literature is concerned with mechanical properties and the effect of deformation on mechanical properties, and most reported orientation gradients occur as a result of deformation. However, to our knowledge, it is the only tool that has previously been used to analyze orientation gradients.

4.2.1 GND Calculations

In order to calculate GND density, we followed the approach of Kysar *et al.* [42], which was to parametrize the Nye tensor in terms of the slip systems of the sample under investigation. When using this analysis it is important to determine which slip systems to use to parametrize the Nye tensor because not all of the slip systems necessarily contribute to the rotations

being analyzed. In the case of α tantalum, which is BCC, there are 12 primary slip systems, the $\{011\}\langle 111 \rangle$ type, and 36 more that are close in energy, 12 $\{112\}\langle 111 \rangle$ type and 24 $\{123\}\langle 111 \rangle$ type. However, it has been shown that for pure tantalum annealed in UHV, such as our films, the primary $\{011\}\langle 111 \rangle$ -type slip systems dominate [82]; this decreases the number of possible slip systems from 48 to 12. Even with 12 slip systems, the system of equations used to solve for GND density is underdetermined because there are 12 unknowns—the GND density on each slip system—but only 9 equations—one per component of the Nye tensor. There was no basis for eliminating any of the 12 primary slip systems, so we applied another technique of Kysar *et al.* and used the L_2 norm to compute GND densities with the lowest strain energy [42].

4.2.2 GND Calculation Results

The results of the GND density calculations for the film shown in Figure 4.1b are shown in Figure 4.2 for two slip systems; Figure 4.2a shows slip system $(101)[\bar{1}\bar{1}1]$ and Figure 4.2b shows slip system $(10\bar{1})[1\bar{1}1]$. These GND density maps are noisy and reveal few, if any, distinguishable features except for high dislocation densities at high-angle boundaries. These are in contrast to GND density maps such as those in Figures 10–12 in “High strain gradient plasticity associated with wedge indentation into face-centered cubic single crystals: Geometrically necessary dislocation densities” by Kysar *et al.* [42]. Kysar *et al.* performed wedge indentation on single-crystal aluminum and copper samples, measured the orientation via EBSD on a cross-section of the indented region, and then calculated GND densities on several different slip systems to account for their observed changes in orientation [42]. The GND density maps by Kysar *et al.* show that their deformed samples have different active slip systems in different regions of the sample, for example in their Figure 10 it can be seen that the $(1\bar{1}1)[\bar{1}12]$ slip system is only active on the right side of the wedge indenter because of the high GND densities there and nowhere else, and in their Figure 12 that the $(\bar{1}11)[1\bar{1}2]$ slip system is only active on the left side of the sample [42]. This information about specific

active slip systems in specific areas of the film can be used for example to learn more about how samples deform. In contrast, the GND density maps in Figure 4.2 do not contain such useful information. These maps do not have any features that clearly distinguish the GND density on different slip systems, so they do not contain useful information about different slip systems. This is because of the noise present in the EBSD orientation data that were used for the GND calculations. The uncertainty of the orientation of any one pixel is $\pm 0.1^\circ$ and the films contain orientation gradients of approximately $4^\circ/\mu\text{m}$ [1], which means that the average change in orientation from one pixel to the next is similar to the error in the measurement. Because the GND analysis only relies on pixel-to-pixel changes, the noise results in very different GND densities even for adjacent pixels. There is another reason that the EBSD data do not yield meaningful information, which is that the spatial resolution of the EBSD data is not sufficient to be able to distinguish individual dislocations. This can be shown using a simple two-dimensional model that assumes all dislocations lie in the same plane and are evenly spread throughout the film. The equation $\rho = \frac{\theta}{bx}$ from Nye [35] can be used to calculate GND density ρ from the orientation gradient θ/x and the magnitude of the Burgers vector b . Our films have orientation gradients of $4^\circ/\mu\text{m}$ and tantalum has a Burgers vector of 2.863 \AA , which from the above equation result in a GND density of approximately $2.4 \times 10^{14} \text{ m}^{-2}$. If these dislocations are evenly spaced throughout the film, they will be approximately 64.5 nm apart. The EBSD data in Figure 4.1 have a 400 nm resolution, which means that individual dislocations cannot be observed from these data.

4.3 Model Microstructure with Smooth Gradients

Understanding the dislocation structure is critical to understanding the phase transformation and this unique orientation gradient microstructure. However, because of the noise and insufficient spatial resolution of our data it is difficult to use the parametrized Nye tensor analysis described above to compute GND density directly from the film data. In order to

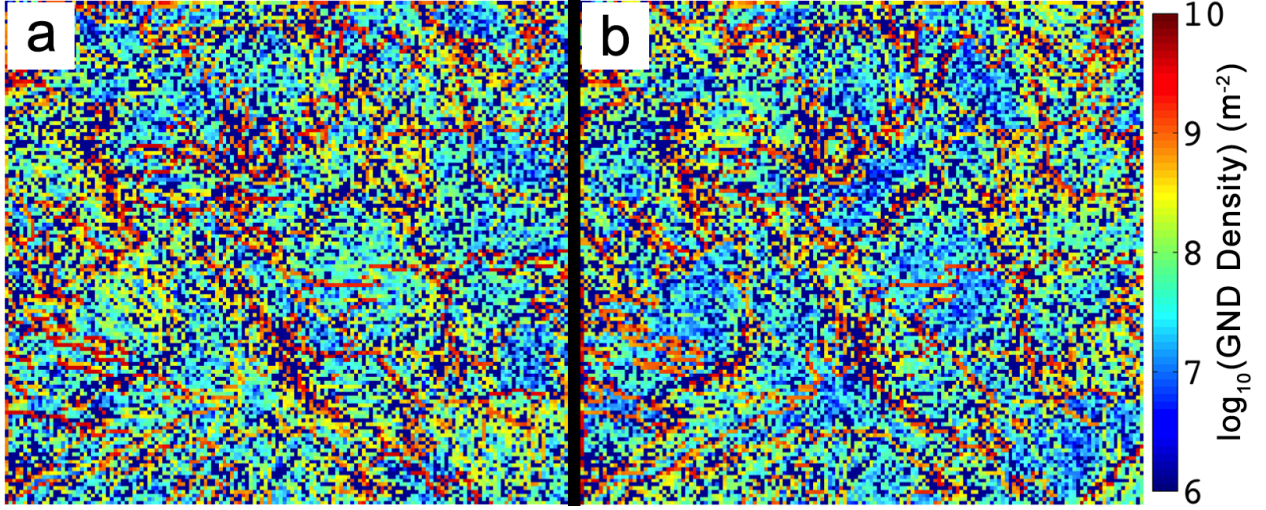


Figure 4.2: GND density maps for (a) $(101)[\bar{1}\bar{1}1]$ and (b) $(10\bar{1})[1\bar{1}1]$ slip systems for the film in Figure 4.1b. These are difficult to interpret primarily because the EBSD orientation data used in the calculations are noisy.

allow for further analysis, we have developed a model microstructure, which we refer to as the smooth model. The smooth model reproduces important aspects of the phase-transformed microstructure and can generate arbitrarily fine orientation data. This permits any analysis that benefits from orientation data that is both smooth and, if desired, with higher spatial resolution than the EBSD orientation data.

4.3.1 Model Development

In order to make model development feasible, we modeled a repeating pattern that we observed in one of the films instead of trying to capture the entire microstructure in one model. The pattern that we modeled is triangular, and shows up in several places in Figure 4.1b, labeled as triangles A, B, and C in Figure 4.3. Triangles B and C are truncated so we focused most of the analysis on triangle A. The repeating triangle that we modeled is as follows: a $\{111\}$ center, surrounded by three alternating regions of $\{001\}$ and $\{011\}$, with the $\{001\}$ regions forming the corners and the 011 regions forming the edges. This triangle is approximately $35\text{ }\mu\text{m}$ wide by $30\text{ }\mu\text{m}$ tall. Some of these triangles have low- and high-angle orientation boundaries (gray and black lines, respectively) that extend outward radially from

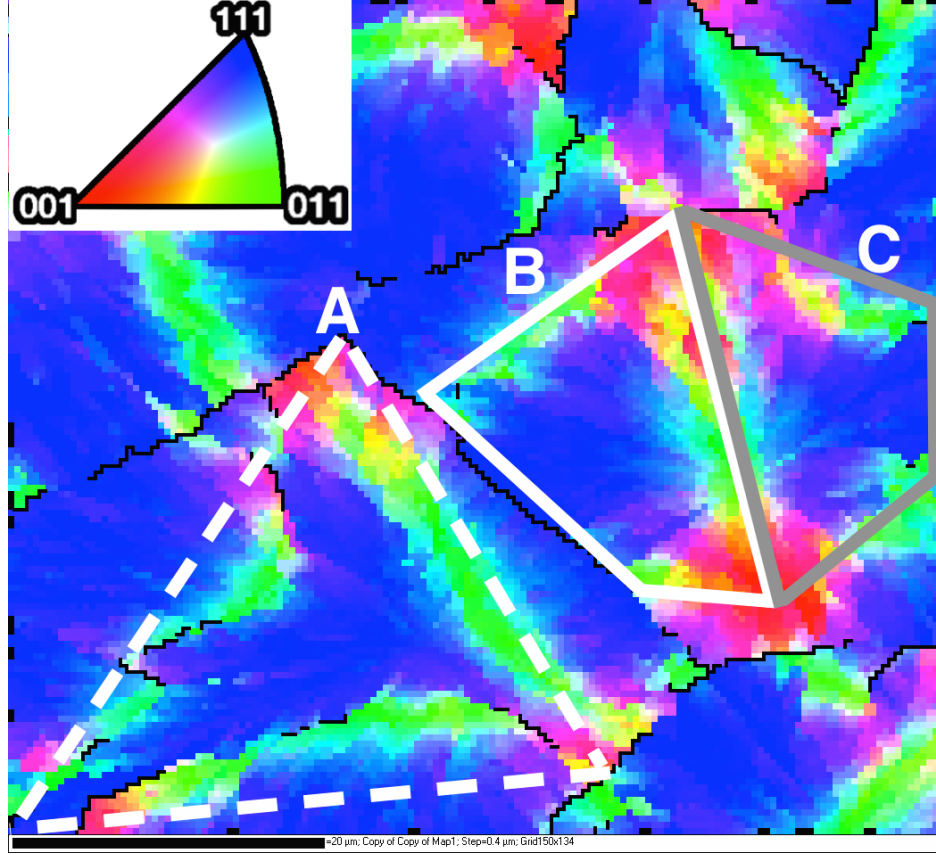


Figure 4.3: Film data with overlaid triangles indicating sections of the film that were modeled. Most of the modeling work was focused on Triangle A, but pole figures of the models of Triangles B and C are shown in Figures 4.9 and 4.10.

the center, indicating that they may have been produced by a radial process.

We had a phenomenological insight for these triangles in the film: if the center of the triangle has the appropriate (111) OOP orientation, then rotating towards the top of the triangle will result in a (001) OOP orientation, and rotating toward the side of the triangle will result in a (011) OOP orientation. These rotations then line up with the triangle in the film, as illustrated in Figure 4.4, where Figure 4.4a shows an illustration of these rotations with the cubes colored by their OOP directions according to the stereographic triangle, and Figure 4.4b shows the region of the film that this matches.

If these rotations are continued in every direction, the result is the orientation map in Figure 4.5a, which shows the OOP component of the orientation colored by the stereographic triangle. Figure 4.5b shows the triangular region of the film that this matches. This sim-

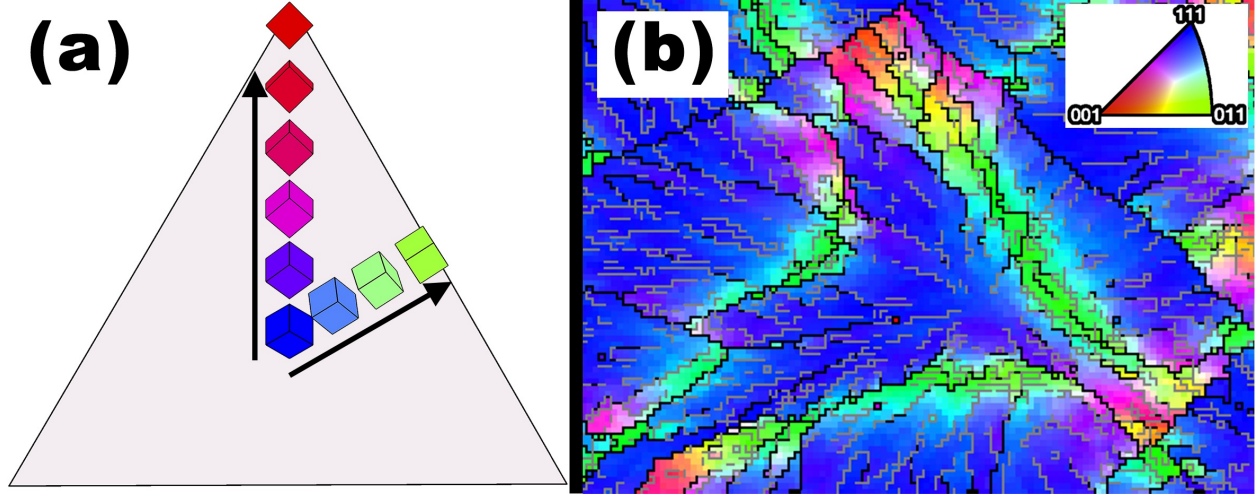


Figure 4.4: Illustration of phenomenological insight that matches triangular regions in the film. (a) is an illustration of a cube with a (111) OOP direction rotated one way resulting in a (001) OOP direction, and rotated another way resulting in a (011) OOP orientation. The cubes are colored by their OOP orientation by the stereographic triangle. (b) is the region of the film that this process matches

ple model produces a good match of symmetry between the two, with each matching the description of the triangle.

A schematic of the simple radial rotation process that produced Figure 4.5a is shown in Figure 4.6. Figure 4.6a shows the selection of a point P at which to compute the orientation, where O is the origin and \vec{r} is the radial vector. Figure 4.6b is an illustration of the actual computation, which involves rotating the starting orientation, represented by the cube, about the axis of rotation \vec{v}_r by an angle ω . The axis of rotation is perpendicular to the radial vector and lies in the plane, and the amount of rotation is proportional to the distance between O and P . These steps are repeated for all points in the triangle.

This simple model matches the symmetry of the film, but does not match the relative sizes of the areas of different orientations, so the magnitude of rotation and the rotation axes need to be modified. Specifically, the $\{011\}$ stripes need to be lengthened, the $\{001\}$ corners need to be shrunk, and the $\{111\}$ center needs to be enlarged. The rotation axes need to be modified to broaden the $\{011\}$ stripes and shrink the $\{001\}$ corners, while the amount of rotation at each point needs to be modified to expand the area of the $\{111\}$ center. The component that modifies the rotation axes is of the form $\theta_{\text{out}} \cos = (\theta_{\text{in}}) + \theta_{\text{in}}$,

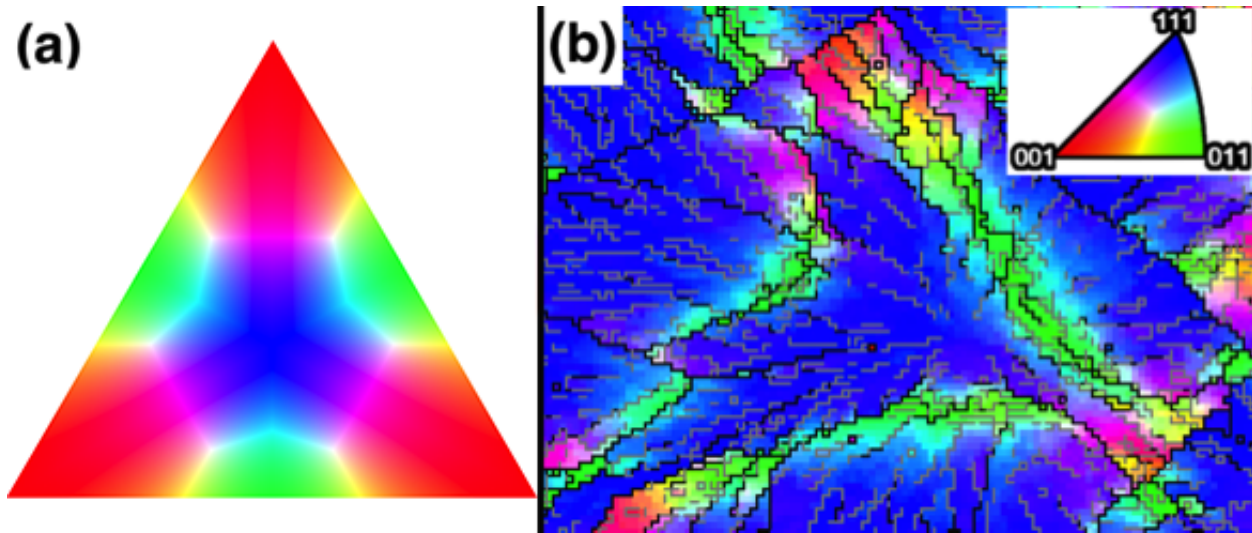


Figure 4.5: (a) Smooth model constant-rate radial rotation map, (b) section of film data from Figure 4.1b that was modeled. Both are colored by OOP orientation according to the inset stereographic triangle. Note the symmetry matches, indicating a radial process yields a good match.

where θ_{in} is the angle of the radial vector in polar coordinates, and θ_{out} is the angle of the vector used to determine the axis of rotation. The component of rotation that modifies the amount of rotation at each point is a complex sigmoidal function. This component is sigmoidal because it allows for slow rotation rates near the center of the triangle, which is all very close to $\{111\}$, smoothly increasing to rapidly rotate through $\{122\}$, and then slow down near the edge of the triangle for a broader 011 region. A similar logic applies to rotating from $\{111\}$ through $\{112\}$ to $\{001\}$ corners. We call the smooth model with these modifications the variable rotation rate model.

4.3.2 Model Validation (Results and Discussion)

The the variable rotation rate model is shown in Figure 4.7, where Figures 4.7a to 4.7c are film data and Figures 4.7d to 4.7f are model data. Black pixels are unindexed portions of the film, and remaining data are colored by the inset stereographic triangle. Figures 4.7a and 4.7d show the OOP orientation ((001) axis), Figures 4.7b and 4.7e show the component of the orientation along the (100) axis, and Figures 4.7c and 4.7f show the component along the

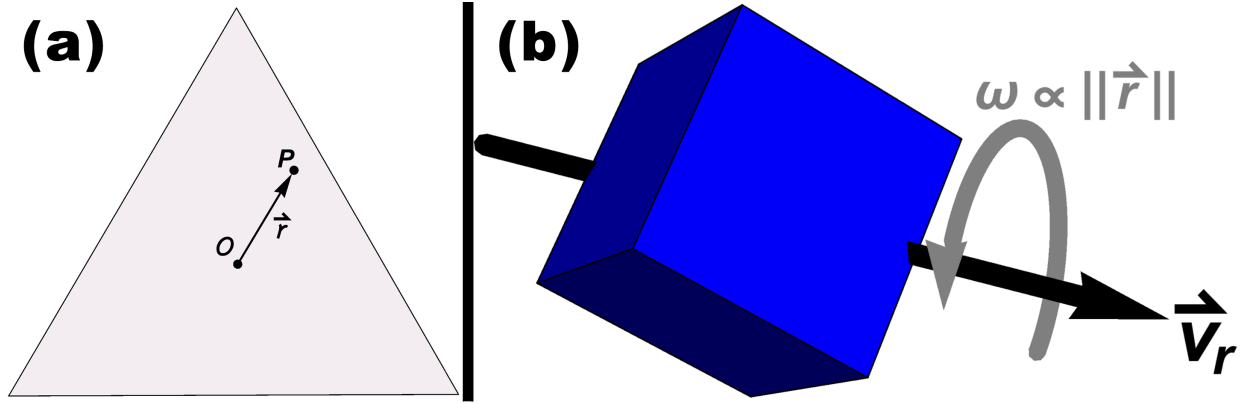


Figure 4.6: Schematic for generating orientation data from smooth model in Figure 4.5. (a) selecting the point at which to calculate the orientation, and (b) illustration of the orientation calculation via rotation, where the cube represents the starting orientation. O is the origin; P is the point at which to compute the orientation; $vecr$ is the radial vector; \vec{v}_r is the axis of rotation, which is perpendicular to \vec{r} and lies in the plane; ω is the amount of rotation and is proportional to the distance between P and O .

(010) axis. In addition to matching the symmetry of the constant rotation rate model, the variable rotation rate model now also matches the spatial distribution of orientations. The match of both symmetry and spatial distribution for both in- and out-of-plane orientation components confirms that the model matches the orientation in the plane of the film as well as out of the plane, for a full three-dimensional orientation match. It is also evident that the OOP component has threefold rotational symmetry. The symmetry is not obvious for the in-plane components of orientation because projecting high-dimensional data into a lower-dimensional representation reduces apparent symmetry.

The orientation data from the smooth model and the corresponding region of the film, Triangle A in Figure 4.3, are shown as pole figures in Figure 4.8. Figures 4.8a to 4.8c are the film data and Figures 4.8d to 4.8f are the model data, and Figures 4.8a and 4.8d are (001) pole figures, Figures 4.8b and 4.8e are (011) pole figures, and Figures 4.8c and 4.8f are (111) pole figures. The match of these pole figures confirms the similarity between the orientation data from the film and that generated by the model. The model can also match other regions of the same film, as indicated by pole figures of triangles B and C, shown in Figures 4.9 and 4.10, respectively. Similarly to Figure 4.8, Figures 4.9a to 4.9c and Figures 4.10a to 4.10c are film data and Figures 4.9d to 4.9f and Figures 4.10d to 4.10f are the

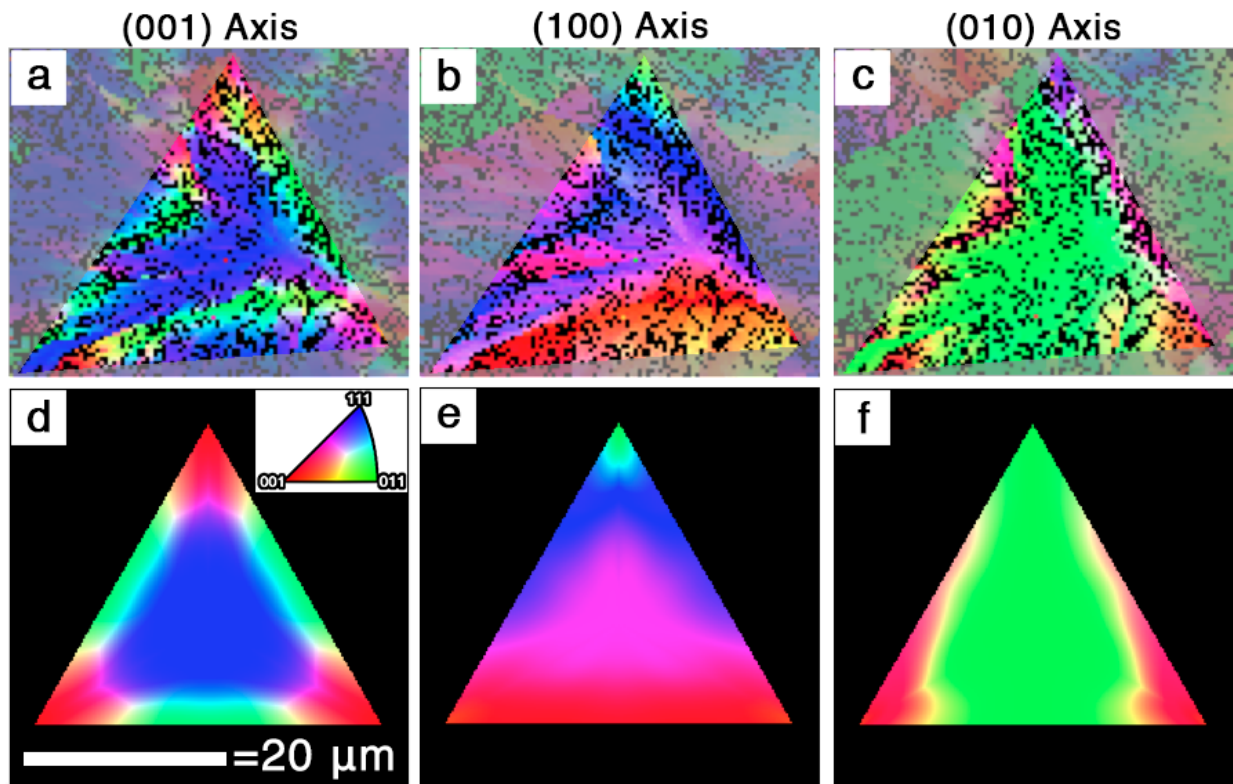


Figure 4.7: (a), (b), and (c) are components along the respective (001), (100), and (010) axes of film orientation data for the region of the film shown in Figure 4.5b, and are colored by the inset stereographic triangle. (d), (e), and (f) are the model data for the same region along the same respective axes. The black pixels in (a–c) are unindexed portions of the film.

model data for their respective regions. Most of the modeling work was focused on Triangle A, but Triangles B and C were modeled by using the appropriate starting orientation to confirm that the smooth model can match more than just one region of the film.

The pole figures also showed that the model captures twin-like behavior that can be seen in Figures 4.9 and 4.10, where the corresponding pole figures appear to be reflected about a line rotated approximately 15° from the vertical. That is, the (111) region at the center of triangle B is related to the (111) material at the center of triangle C by a reflection operation. It is as if they were twins of each other except that this twin-like behavior is not a result of actual twinning or a mirror operation, but is a consequence of the same rotation scheme with different in-plane starting orientations that are a mirror of one another about the twin-like axis. In the (111) pole figures in Figures 4.8c, 4.9c, and 4.10c it can be seen

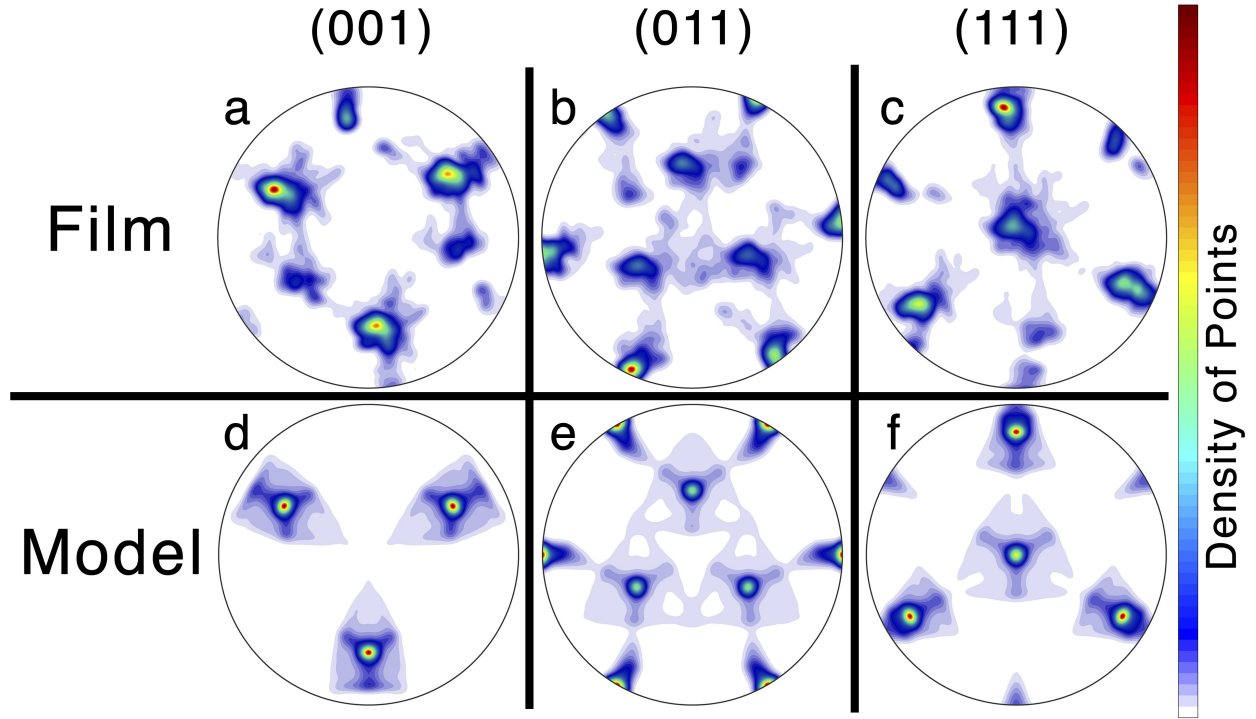


Figure 4.8: Pole figures of film and model data from Triangle A in Figure 4.3. (a–c) are film data and (d–f) are model data. (a) and (d) are (001) pole figures, (b) and (e) are (011) pole figures, and (c) and (f) are (111) pole figures.

that the high-density regions near the periphery of the pole figure have small regions directly opposite them at the other edge of the pole figure with a low to moderate density of points. This is not a separate population of $\{111\}$ grains rotated 60° in the plane of the film but rather an effect of the orientation gradient. As the orientation changes and one of the $\{111\}$ poles crosses over the edge (*i.e.*, dips below the base plane of the pole figure), another one emerges directly opposite it; this is easily demonstrated using a physical cube.

The orientation data in Figure 4.7 and the pole figures in Figures 4.8, 4.9, and 4.10 illustrate different things, neither of which can be learned from the other. The orientation maps show the spatial distribution of the orientations and orientation gradients, while the pole figures provide complete three-dimensional orientation data.

Calculated GND density maps for the smooth model data are shown in Figure 4.11, where Figure 4.11a shows slip system $(101)[\bar{1}\bar{1}1]$ and Figure 4.11b shows slip system $(10\bar{1})[1\bar{1}1]$. Even with the smooth model data, however, the calculated GND density maps were still

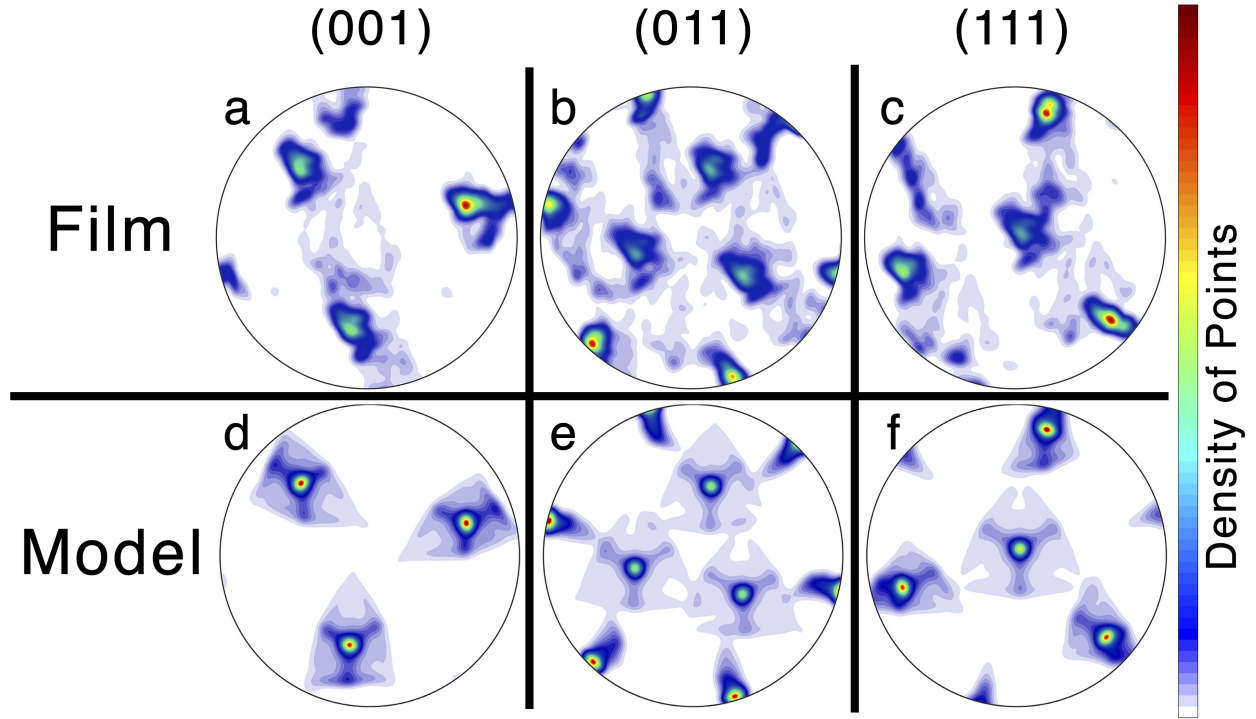


Figure 4.9: Pole figures of film and model data from Triangle B in Figure 4.3. (a–c) are film data and (d–f) are model data. (a) and (d) are (001) pole figures, (b) and (e) are (011) pole figures, and (c) and (f) are (111) pole figures.

not useful for analysis. Using the smooth model data has removed the noise present in Figure 4.2, but has introduced another problem, which is that the rotations in the model are not associated with specific physical dislocations so they do not correspond to BCC slip systems used in the Nye tensor analysis. This means that when the rotations are parametrized in terms of slip systems, there may be multiple slip systems that can account for each rotation equally well so the resulting GND densities on each slip system can change depending on the order in which they are parametrized. This causes the fine details of the GND density maps in Figure 4.11, specifically the low-density loops, to change depending on the order in which the slip systems are parametrized. This means that these results cannot be used to distinguish low- and high-GND density regions on each slip system, which would tell us more about the dislocations required for this microstructure. Thus, in order for useful Nye tensor analysis, the following are needed: smoothly changing orientation data, rotations that correspond to physical slip systems, and sufficient spatial resolution to detect

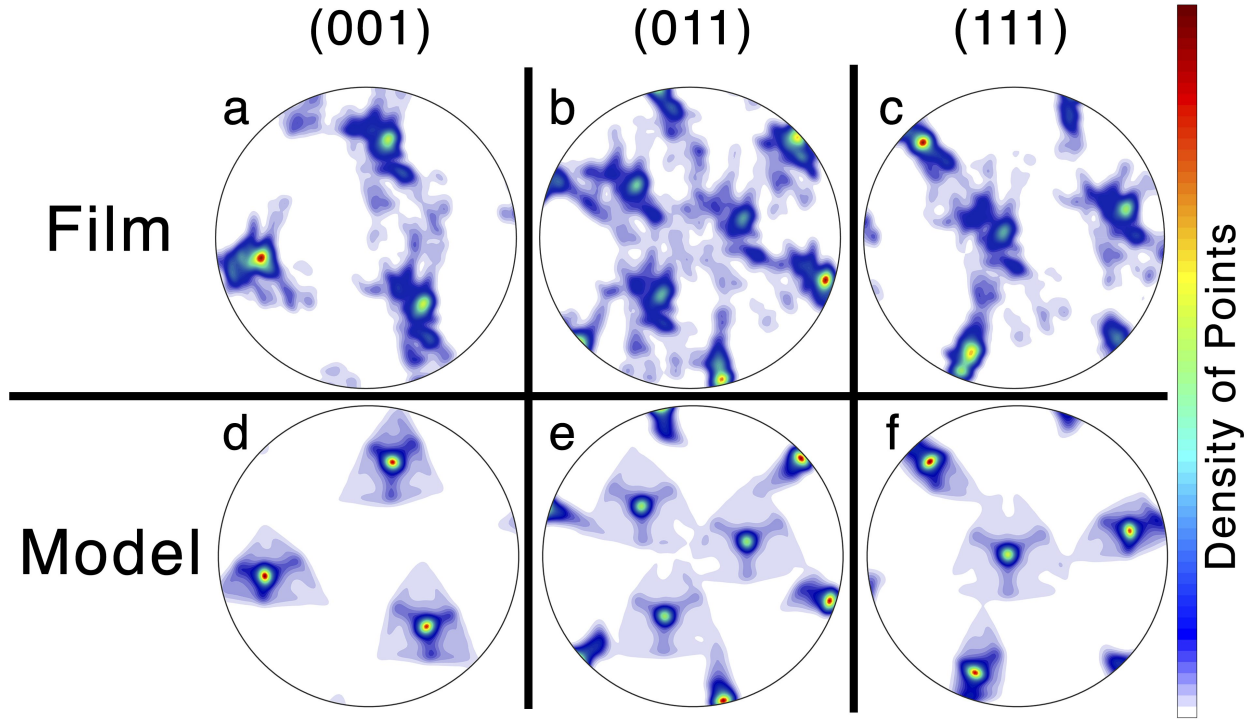


Figure 4.10: Pole figures of film and model data from Triangle C in Figure 4.3. (a–c) are film data and (d–f) are model data. (a) and (d) are (001) pole figures, (b) and (e) are (011) pole figures, and (c) and (f) are (111) pole figures.

regions that have high GND densities on one slip system and low GND densities on another in order to obtain useful information about the required dislocations. It was not feasible to have all of these for the film or model data, so we took the opposite approach, generating dislocation structures using a genetic algorithm and calculating how well the orientation data they produced matched the smooth model.

4.4 Genetic Algorithm Model

Genetic algorithms (GAs) are an optimization technique that mimics evolution by incorporating the principles of natural selection, inheritance of genetic material, crossover, and mutation. Candidate solutions, called individuals, pass down their genetic material, represented as parameter encodings, to a number of offspring based on how fit each individual is. Each individual is a potential solution generated by the algorithm, and all of the individuals

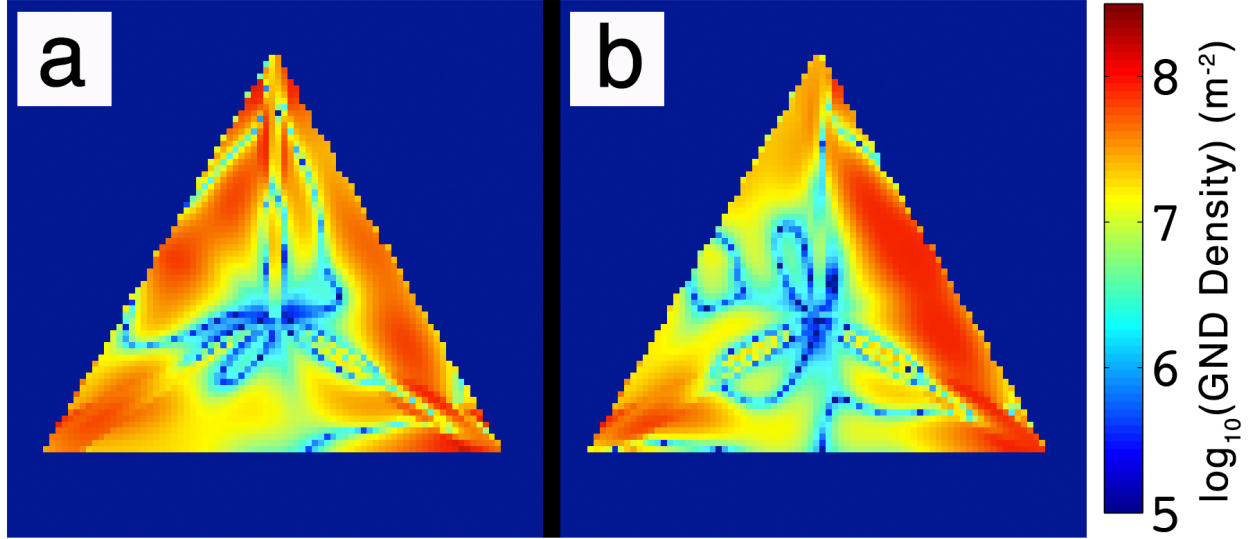


Figure 4.11: GND density maps for (a) $(101)[\bar{1}\bar{1}1]$ and (b) $(10\bar{1})[1\bar{1}1]$ slip systems for the model triangle. Using the smooth model data has removed the noise in Figure 4.2, but still does not yield a useful GND density map.

are collectively referred to as a population. Each iteration of the algorithm produces a new population, and the population from a particular iteration of the algorithm is called a generation. Each individual is evaluated according to a fitness function that determines how fit that individual is. Examples of criteria that may be used in a fitness function include cost, weight, and efficiency. It is possible to evaluate fitness based on multiple criteria, and an algorithm that does this is called a multi-objective genetic algorithm [79]. An individual with a high fitness will produce more offspring than one with a low fitness, and, as in evolution, this ensures that beneficial genes are more likely to be passed down from one generation to the next and spread throughout the population. The process by which individuals produce offspring is called crossover, which combines the genetic material from two individuals. Another feature of GAs is mutation, which makes small, random changes to the encoding of an individual. This introduces new genetic material to the population, which helps prevent premature convergence of the algorithm.

GAs have several advantages over other optimization techniques. In particular: they are less likely than other techniques to converge prematurely; they can work well with an unintuitive and complex solution space; they are inherently parallelizable; and they tend

to produce robust solutions. GAs are less likely to converge prematurely because, if implemented properly, they start with and maintain a diverse population, which allows them to explore a larger area of the search space. The problem of dislocation structure generation satisfies all of the informal criteria typically used to determine if a GA is the best approach for a particular problem. These criteria include a complex, nonintuitive solution space with variables that interact in unknown ways; a good way to encode and evaluate potential solutions; and a need for a sufficiently good solution but not the absolute best. The solution space of this problem, which will be discussed in more detail later, is complex and unintuitive because it is difficult to determine the effect that changing the shape and Burgers vector of a dislocation will have on the resulting orientation map without performing the computation. Optimizing these orientation maps against dislocation density is also a difficult problem. For this problem it is easy to construct a good fitness function, such as a combination of the average misorientation between the orientation map produced by a particular set of dislocations and that of the smooth model from the previous section and the total dislocation density. It is also relatively straightforward to encode potential solutions; for example, we may store each dislocation as an array of (x, y) coordinates and its Burgers vector as an array with three elements. It is not likely that this problem has a unique solution, *i.e.* it is probable that there are multiple different dislocation configurations that yield indistinguishably good orientation maps and that have comparable dislocation densities; thus it is not necessary to find the absolute best solution. Additionally, it would not necessarily be clear if one of these indistinguishably good solutions were found because we do not have a means of verifying that a particular solution cannot be improved.

4.4.1 Details and Methods

The genetic algorithm model was written in Python, taking advantage of available software packages to improve both development time and run time. In particular, the Distributed Evolutionary Algorithms in Python, or DEAP, package [85] provides an excellent framework

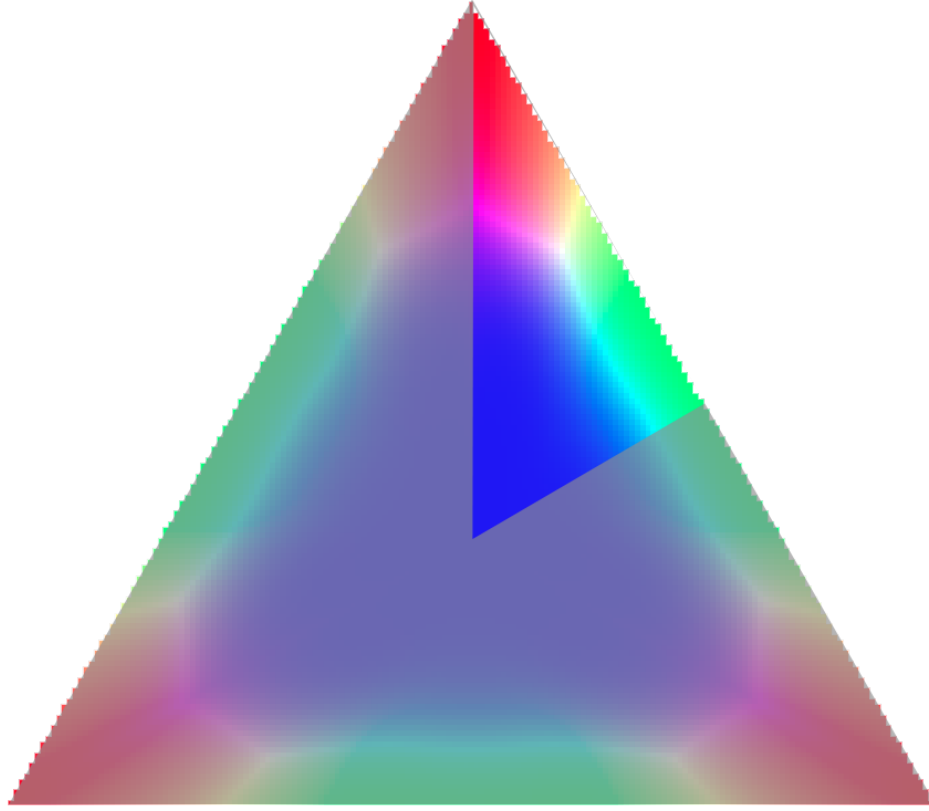


Figure 4.12: Due to its threefold rotational symmetry, this one-sixth of the smooth model triangle is all that needs to be modeled, allowing more detail and reducing computation time.

for creating GAs, and the Scalable COncurrent Operations in Python, or SCOOP, package [86] enables easy parallel and distributed computation. The standard NumPy [87], SciPy [88], and Matplotlib [89] packages were also used.

A key feature of the smooth model that can be exploited is its symmetry. This is illustrated in Figure 4.12, where the outline indicates the area that was modeled by the genetic algorithm model. Modeling only one-sixth of the triangle allows the model to have more detail and require less computation time. This smaller triangle is approximately 8 μm wide and 18 μm tall.

4.4.1.1 Individual Encoding

A key aspect of any GA is parameter encoding, as this determines how mutation and crossover will work and thus how the population will evolve. Our genetic algorithm had

seven parameters for each individual on which crossover and mutation could operate.

The first parameter is the number of in-plane (IP) dislocations. The IP dislocations lie in the plane of the film and account for a change in orientation along the radial direction. This is stored as an integer.

We chose to model the IP dislocations as all lying in the same plane, because we only have orientation data from the surface of the film (*i.e.* we neglect orientation gradients in the film thickness direction). To avoid complications that would arise due to dislocation interactions and reactions, the model is set up so that the IP dislocations are prevented from crossing one another, but are still allowed to change shape in order to optimize the resulting orientation. In order to achieve this, we used the level set method [90] to determine dislocation coordinates. A level set L for a function f of n variables is the set of all points (x_1, \dots, x_n) for which the function at that point $f(x_1, \dots, x_n)$ is equal to some value c , mathematically represented as $L(f) = \{ (x_1, \dots, x_n) \mid f(x_1, \dots, x_n) = c \}$. In the case that there are only two variables, a level set is called a contour. A function of only two variables only has one value at any point, meaning that its contours do not intersect. For this reason we used contour lines to represent dislocations. We used evenly spaced contour lines, so we predicted some of the properties that the level set function would need to have in order to produce contour lines to account for the rotations in the model. We made these predictions in order to try to seed the GA solutions in an area of the solution space that would be as close to the optimal solution as possible. The function should be steep where the orientation changes rapidly—when passing from (1 1 1) through (1 2 2) to (0 1 1), for example—so that the contour lines are closer together, and nearly flat where it does not—in the center of the triangle, for example—so that the contour lines are farther apart. The component of the smooth model that determines the amount of rotation at each point has some of these features, thus we used this component as a basis for the level set function. The algorithm could then add noise to this in order to allow the dislocations to change shape to find the configuration that resulted in the best orientation match with the smooth model. The

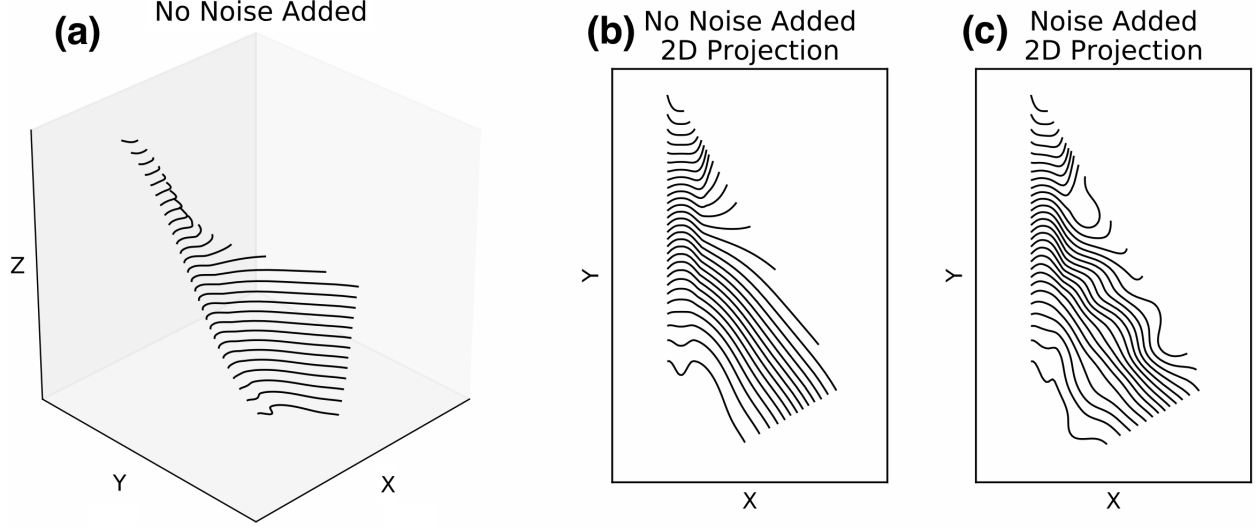


Figure 4.13: (a) 3D and (b) 2D views of the contour lines from the level set function with no noise added; (c) 2D view of contour lines with noise added. The use of contour lines to represent dislocations ensures that they do not cross while still allowing for their shape to change.

contour lines from the level set function are shown in Figure 4.13, where Figure 4.13a shows the contour lines in three dimensions, Figure 4.13b shows a two-dimensional projection of the same contour lines, and Figure 4.13c shows contour lines from the level set function with noise added. In these figures, the x and y coordinates are the position in the film and the z coordinate is the height of the level set function. For each contour line we want to obtain we assign a height h of the level set function and compute the contour line by taking the set off all (x, y) points where the level set function is equal to that height, *i.e.* $\{ (x, y) \mid z(x, y) = h \}$.

The second parameter is the number of noise points added to the level set function. In order to allow the algorithm to change the shape of the level set function, we allowed it to add noise in the form of two-dimensional Gaussian curves to the level set function at random locations. The number of noise points was stored as an integer.

The third parameter is the locations of the noise points. The locations where noise was added to the level set function were stored in a $N_{NP} \times 2$ array, where N_{NP} is the number of noise points. Each row of the array contained coordinates (n_x, n_y) where the noise point was to be added to the level set function. For each point (n_x, n_y) where noise was to be added

to the level set function, the Gaussian

$$\Delta z = \frac{1}{2\sqrt{2\pi}} \exp\left(-\frac{(x - n_x)^2 + (y - n_y)^2}{2}\right) \quad (4.1)$$

was used, where Δz is the change in height due to the added noise and (x, y) correspond to coordinates in the level set array. The maximum height of each Gaussian is very small relative to the height of the level set function so the changes in height due to each noise point are small as well. We used a Gaussian because its shape ensures a smooth, continuous change in the function so the contour lines do not have any kinks, which could happen if noise were added at a single point to the function without some distribution. Typically, the shapes of the contour lines do not change much from one generation to the next. This is for two reasons, which trade off in importance as more generations are computed. Early in the algorithm they do not change much because each individual noise point does not have a very large effect on the shapes of the contour lines and the noise points are randomly spread throughout the triangle so the effect of each noise point is isolated. Later in the algorithm, as the locations of the noise points begin to converge, they are no longer isolated. Because they are close together they start having a larger effect, but this is balanced by the fact that the remaining individuals are more similar so the points are more and more likely to come from a similar location.

The fourth parameter is the list of Burgers vectors for each IP dislocation. Each IP dislocation has a $\langle 111 \rangle$ -type Burgers vector because that is the lowest energy type of dislocation in a BCC material. This parameter is stored as an $N_{IP} \times 3$ array where N_{IP} is the number of IP dislocations. Each row of the array is the Burgers vector for the corresponding IP dislocation. Each contour line is then assumed to represent multiple physical dislocations, which we call a bundle. Bundling dislocations together in the model serves both to reduce computational complexity and to ensure that the results do not have so many dislocation lines that they become difficult to distinguish. We found that the optimal bundle size for

the IP dislocations was 25, so each contour line represented 25 physical dislocations. The number of IP dislocations was not changed by adding noise, so each IP dislocation retained the same Burgers vector because the IP dislocations were always assigned Burgers vectors in the same order from the list, *e.g.* the first IP dislocation is always taken at the lowest contour height and is in the bottom left of the triangle, and corresponds to the first Burgers vector in the array of IP dislocation Burgers vectors. The orientation of the Burgers vectors are referenced to the crystal orientation at each point, so for example a $[1\ 1\ 1]$ Burgers vector at the bottom left of the triangle would point directly out of the page, and a $[1\ 1\ 1]$ Burgers vector at the top of the triangle would point straight up ($\theta = 35.3^\circ$ and $\varphi = 90^\circ$ in spherical coordinates, where θ is the polar angle and φ is the azimuthal angle) and out of the page at a $\sim 54.7^\circ$ angle.

The fifth parameter is the number of out-of-plane (OOP) dislocations. OOP dislocations are perpendicular to the plane of the film, and allow us to eliminate what would otherwise be elastically stored strains between pixels rotating perpendicularly to the radial direction. This parameter is stored as an integer.

The sixth parameter is the locations of the OOP dislocations. Because the OOP dislocations were modeled as being perpendicular to the plane of the film, each of their locations were represented as a single point in the model. This parameter was stored as an $N_{OOP} \times 2$ array, where N_{OOP} is the number of OOP dislocations and each row of the array contained the (x, y) coordinates of that OOP dislocation.

The seventh parameter was the Burgers vector of each OOP dislocation. The OOP dislocations also had $\langle 1\ 1\ 1 \rangle$ -type Burgers vectors. This parameter was stored in a $N_{OOP} \times 3$ array where N_{OOP} is the number of OOP dislocations. Each row of the array is the Burgers vector for the corresponding OOP dislocation. Each OOP dislocation also represented multiple physical dislocations, and we found that the optimal bundle size for OOP dislocations was 3, so each OOP dislocation represented 3 physical dislocations.

4.4.1.2 Fitness Function

Our genetic algorithm model uses two metrics to determine the fitness of each individual: average misorientation between the orientation produced by its dislocations and the orientation from the smooth model, and total dislocation density. If only average misorientation is used and dislocation density is not, the algorithm will rarely remove dislocations, and those with no effect or only a very small one will accumulate, resulting in unphysically high dislocation densities. After determining the orientation at each pixel, the misorientation was calculated at each pixel and was converted to axis-angle form in a way that always returns a non-negative angle. The average of these misorientation angles was computed across the entire map. The dislocation density was calculated as the sum of the lengths of all IP and OOP dislocations, taking into account the bundle size for each dislocation type, divided by the volume of the triangle, computed as the area of the triangle times the thickness of the film (650 nm). A reasonable range for the relative weight between average misorientation and dislocation density is between 1:3 and 3:1 in units of degrees to 10^{14} m^{-2} . This yielded a dual-objective problem, as both the misorientation and dislocation density were minimized. While it is often popular to use a pareto-optimal algorithm such as the Non-dominated Sorting Genetic Algorithm (NSGA) II [91], we chose to simply combine the two metrics with a relative weight and use that as the fitness metric, and minimized that metric in order to maximize fitness. Within a reasonable range of weights, the relative weight of misorientation and dislocation density did not have a particularly large effect, and combining the two into a single metric simplified the algorithm. Tournament selection was used to determine which individuals would reproduce to make the next generation because it is algorithmically simple, straightforward to parallelize, and allows for easy adjustment of selection pressure.

4.4.1.3 Initialization, Selection, Crossover, and Mutation

Each of parameter was initialized randomly within a particular range because the algorithm was observed to almost always converge on a value within that range across multiple runs,

and setting the parameters to a value outside the range resulted in a longer convergence time. The number of IP dislocations (contour lines) was randomly selected between 75 and 150. The Burgers vector for each IP dislocation was initially chosen randomly from the eight $\langle 111 \rangle$ directions during model development, but it became apparent that the algorithm was selecting two specific Burgers vectors, $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$, so we modified the algorithm to randomly initialize one of those two Burgers vectors to each IP dislocation in order to reduce convergence time. This is discussed in more detail in Sections 4.4.2 and 4.5. The number of points of noise added to the level set function was randomly selected between 20 and 75. The locations of the noise points were randomly selected to be anywhere inside the triangle in order to allow the algorithm as much freedom to change the shape of the IP dislocations as possible. The number of OOP dislocations was randomly selected between 100 and 1000. We used such a wide range because the converged value changed a lot from generation to generation. This is discussed in more detail in Sections 4.4.2 and 4.5. The locations of the OOP dislocations were randomly selected to be anywhere inside the triangle because the algorithm did not converge to a structure with the OOP dislocations in a particular part of the triangle. The Burgers vectors for the OOP dislocations were randomly selected from all eight $\langle 111 \rangle$ directions because they did not converge on any particular Burgers vector.

Tournament selection was used to determine which individuals would reproduce to make the next generation because it is algorithmically simple, straightforward to parallelize, and allows for easy adjustment of selection pressure.

Crossover operated on four of the seven parameters: the Burgers vectors of the IP dislocations, the locations of the noise points, the locations of the OOP dislocations, and the Burgers vectors of the OOP dislocations. Crossover operated by randomly selecting between 35 and 65% of the genetic material for each parameter to exchange between individuals. This percentage was of whichever individual had the smaller amount of that parameter, for instance when performing crossover on the noise points it would select between 35 and 65% of whichever individual had fewer noise points. For each parameter on which crossover was

performed, it randomly selected a contiguous region of the array of that parameter, of the size determined earlier, from each individual and exchanged them. For example, crossover is being performed on two lists A and B , where A has i elements (A_1, \dots, A_i) , B has j elements (B_1, \dots, B_j) , and $i < j$. Since i is smaller than j the algorithm randomly generates an integer value p between 35% and 65% of i . It then randomly selects p contiguous points from A and B by generating random numbers C_A between 1 and $i - p$ and C_B between 1 and $j - p$. It then takes the elements from A between C_A and $C_A + p$ and swaps them with elements C_B through $C_B + p$ from B . Crossover was performed separately on the list of Burgers vectors of the IP dislocations and the list of noise points, but for the locations of the OOP dislocations and the list of their Burgers vectors the same elements in the different lists were swapped (so that the OOP dislocations kept the same location and Burgers vector after crossover). This is discussed in more detail in Sections 4.4.2 and 4.5. Crossover did not change the number of IP dislocations, the number of noise points, or the number of OOP dislocations; those parameters were modified by mutation.

Mutation operated on all seven parameters, and each individual had a 25% chance of being mutated. The possible changes for mutation of each parameter are as follows: the number of IP dislocations could randomly increase or decrease by up to 5; up to 15 IP dislocations could have their Burgers vector randomly reassigned; the number of noise points could randomly increase or decrease by up to 10; up to 10% of the noise points could have their coordinates changed by up to 1 μm in the x or y direction; the number of OOP dislocations could increase or decrease by up to 50; and up to 50 OOP dislocations could have their Burgers vectors randomly reassigned.

4.4.2 Model Validation (Results)

Figure 4.14 shows a good result produced by the genetic algorithm model. Figure 4.14a shows the reference OOP orientation colored by the stereographic triangle from the smooth model data, Figure 4.14b shows the orientation produced by the dislocations from that individual,

colored by OOP orientation, Figure 4.14c shows the misorientation between Figure 4.14a and Figure 4.14b, colored by the included color bar, and Figures 4.14d and 4.14e show the IP and OOP dislocations colored by their Burgers vectors. This individual has a fitness metric of 4.84, an average misorientation of 1.39° , and a dislocation density of $3.45 \times 10^{14} \text{ m}^{-2}$. The low average misorientation means it is a very close match to the smooth model, which can be seen in the similarity between Figures 4.14a and 4.14b. Most of the IP dislocations are $[\bar{1}\bar{1}1]$, which are white in Figure 4.14d, with some $[\bar{1}11]$, which are red in Figure 4.14d. When run multiple times, the genetic algorithm model often produces individuals with $\sim 80\%$ $[\bar{1}\bar{1}1]$ Burgers vectors and almost all the rest $[\bar{1}11]$ Burgers vectors for the IP dislocations, and when it does not the match is not as good. The results from 3 separate runs are shown in Figure 4.15, where Figures 4.15a to 4.15e, Figures 4.15f to 4.15j, and Figures 4.15k to 4.15o are each the best individual from their respective run. For each individual, the first plot is the orientation data from the smooth model, colored by OOP orientation, the second is the orientation data produced by the dislocations of that individual, colored by OOP orientation, the third is the misorientation between the first two, and the last two show the IP and OOP dislocations colored by their Burgers vectors. From examination of these results it appears that the IP dislocations with $[\bar{1}\bar{1}1]$ Burgers vectors result in a match for most of the triangle except for the lower right corner, which the IP dislocations with $[\bar{1}11]$ Burgers vectors help match.

Across multiple runs the genetic algorithm model converges on a similar pattern of IP dislocations, which can be seen in Figures 4.15d, 4.15i, and 4.15n. In this pattern, most of the IP dislocations are smooth except near the middle left side of the triangle, where there are approximately 5 to 10 dislocations that change direction several times to form loops and swirls. Each population of dislocations with different Burgers vectors tends to be located in the same region in the triangle, with most of the $[\bar{1}11]$ dislocations concentrated in the lower left portion of the triangle, typically mixed with some $[\bar{1}\bar{1}1]$ dislocations, and a few $[\bar{1}11]$ near the top of the triangle as well.

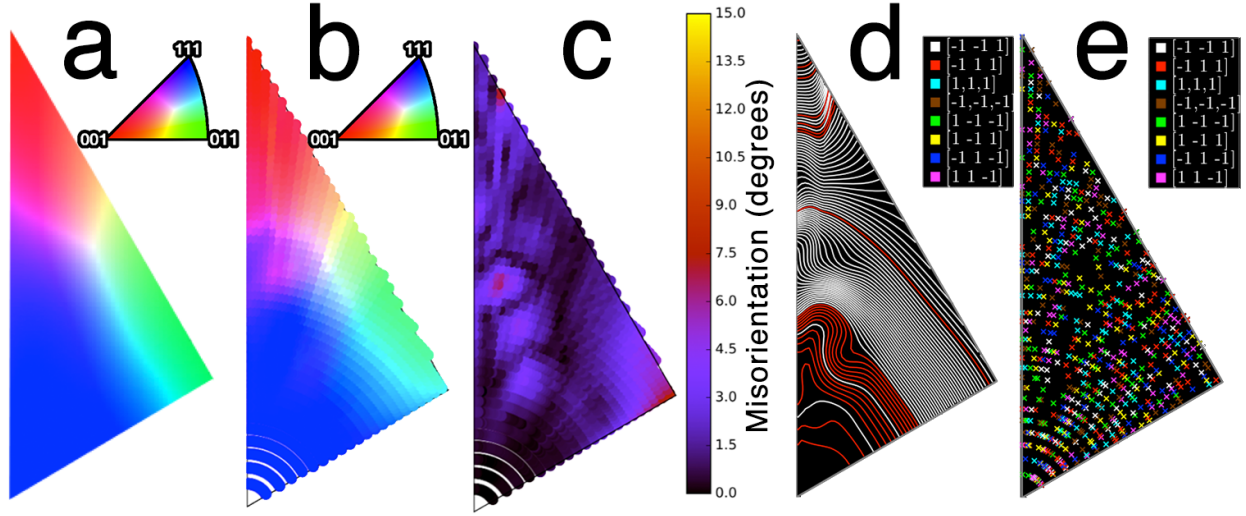


Figure 4.14: Orientation comparison between the smooth model and the genetic algorithm. (a) is the orientation data from the smooth model, (b) is orientation data from the genetic algorithm, (c) is the misorientation between (a) and (b), and (d) and (e) are the in-plane and out-of-plane dislocations, respectively, colored by Burgers vector. This individual has a fitness metric of 4.84, an average misorientation of 1.39° , and a dislocation density of $3.45 \times 10^{14} \text{ m}^{-2}$.

Figure 4.16 shows a multi-generation statistics plot for a run of the genetic algorithm model. Figure 4.16a shows the misorientation, dislocation density, and overall fitness metric of each generation, with misorientation in degrees and dislocation density in 10^{14} m^{-2} . Figure 4.16b shows the number of IP and OOP dislocations, as well as the number of points at which noise was added to the level set function. The misorientation in Figure 4.16a decreases rapidly for approximately the first 40 generations, and then decreases slowly and does not change much past approximately 250 generations, while the dislocation density generally decreases for the first 100 generations and then doesn't change significantly after that. These are typical patterns for the misorientation and dislocation density, although the number of generations that the algorithm takes to converge varies between approximately 150 and 600. In Figure 4.16b, the number of noise points and the number of contour levels do not change significantly throughout the run, which is typical. However, the number of OOP dislocations changes substantially throughout the run. It is common for the number of pole dislocations to change significantly over the course of a run, but this does not always happen.

The relative weight of misorientation and dislocation density in the overall fitness metric

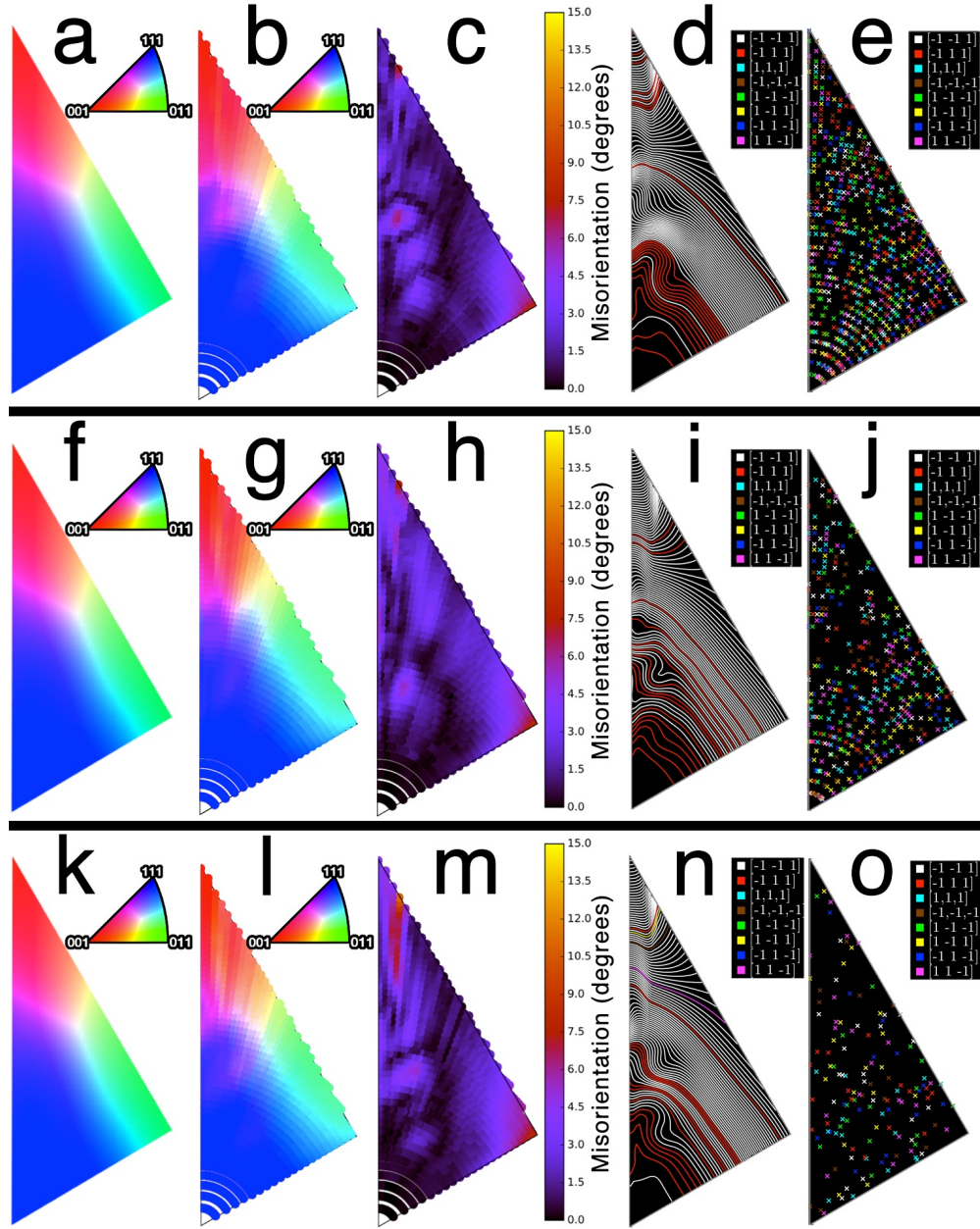


Figure 4.15: Results from several different runs of the algorithm; each is the best individual in its run. For each result, the first image is the orientation from the smooth model, the second image is the orientation map that is generated from the set of dislocations of an individual, the third is the misorientation between the first two, and the third and fourth are IP and OOP dislocations, respectively, colored by Burgers vector. (a–f) is the same as Figure 4.14. (f–j) has fitness metric 4.80, average misorientation 1.59° , dislocation density $3.21 \times 10^{14} \text{ m}^{-2}$. (k–o) has fitness metric 4.89, average misorientation 1.61° , dislocation density $3.28 \times 10^{14} \text{ m}^{-2}$. The algorithm converges on a similar set of IP dislocations each run, with the majority ($\sim 80\%$) $[\bar{1}\bar{1}1]$ dislocations and almost all of the rest $[\bar{1}11]$ dislocations. The different types of dislocations converge to similar areas of the triangles, as well as similar shapes

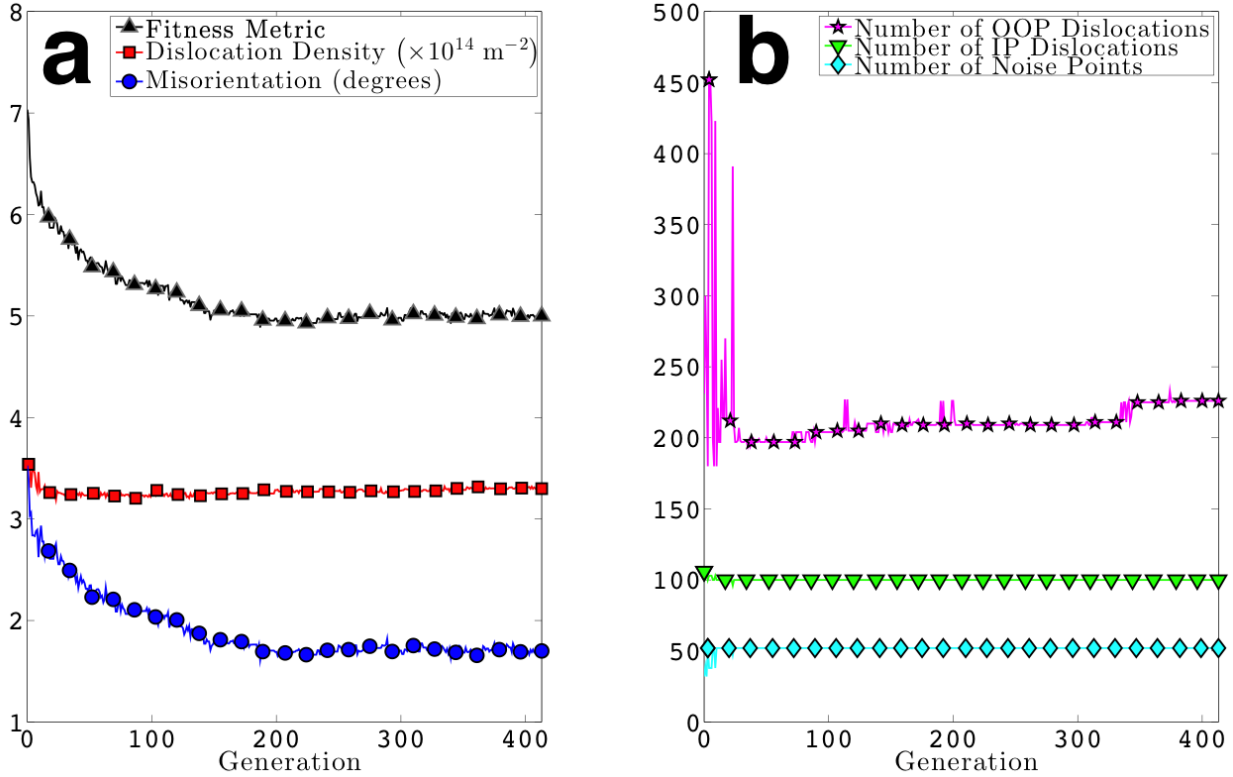


Figure 4.16: Multi-generation statistics plot for a single run of the genetic algorithm. This tracks the following metrics from the best individual from each generation: (a) average misorientation and dislocation density; (b) the number of noise points, the number of IP dislocations, and the number of OOP dislocations.

does not have a large effect, provided that it is within a the range discussed in Section 4.4.1.2. Due to the construction of the fitness metric, the relative weight of dislocation density and misorientation can change as the run progresses, because the misorientation typically has a much larger relative change throughout the duration of a run than does the dislocation density. The average misorientation typically begins near 4° for the first generation, but eventually falls by approximately half, to below 2° . In contrast, the dislocation density rarely changes by more than 10% throughout a run. This means that throughout the run the relative weight of the dislocation density slowly increases.

Neither the mutation rates nor the starting parameters have a significant effect on the fitness of the population after convergence of the algorithm, as they only affect how long the algorithm takes to converge and not the fitness of the converged population. The three most

important mutation parameters are the number of pole dislocations, their Burgers vectors, and the locations where noise is added. The other three parameters—the number of noise points, the number of IP dislocations, and the Burgers vectors of the IP dislocations—do not change very much throughout a particular run regardless of their mutation rates. The starting parameters have the biggest effect on whether the genetic algorithm model converges within several hundred generations instead of many thousands. Of the starting parameters, setting the initial Burgers vector for all the IP dislocations to $[\bar{1}\bar{1}1]$ will make it converge the most quickly; anything else will cause the algorithm to take more time, but will not make the converged solution any worse. Starting the algorithm with too many or too few IP dislocations, OOP dislocations, or noise points will similarly make the genetic algorithm model take longer to converge but will ultimately not affect how good a match there is once it does converge; as well, the algorithm typically converges on similar values each run. The locations of the noise points also tend to converge on similar locations, specifically the center left of the triangle where the IP dislocations change direction several times near the edge of the triangle, as can be seen in Figures 4.15d, 4.15i, and 4.15n. In summary, the starting configuration and mutation parameters do not have much of an effect on the fitness or similarity of the dislocation structure after the population converges, but only affect how many generations are required for the algorithm to converge.

4.5 Discussion

The $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ IP dislocations are very prevalent, and we observed that when the algorithm created individuals with other IP Burgers vectors the misorientation was worse. As a way to determine how necessary these Burgers were, we ran the algorithm first using only those Burgers vectors for the IP dislocations and then again with those Burgers vectors excluded (*i.e.* using only the other 6 $\langle 111 \rangle$ -type Burgers vectors). The result from running the algorithm with only the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors is shown in Figure 4.17, where

Figure 4.17a shows the orientation data of the smooth model colored by OOP orientation, Figure 4.17b shows the orientation data produced by the dislocations of this individual, colored by OOP orientation, Figure 4.17c shows the misorientation between them, and Figure 4.17d shows the dislocations from this individual. This individual has a fitness metric of 5.08, an average misorientation of 1.36° , and a dislocation density of $3.72 \times 10^{14} \text{ m}^{-2}$. This excellent match confirms that the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors are adequate, and that the other 6 $\langle 111 \rangle$ -type Burgers vectors are not necessary for the IP dislocations. We then ran the algorithm without the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors for the IP dislocations, but allowed the other 6 Burgers vectors. The results from that run are shown in Figure 4.18, where Figure 4.18a is the orientation data from the smooth model colored by OOP orientation, Figure 4.18b is the orientation produced by the dislocations of this individual, colored by OOP orientation, Figure 4.18c is the misorientation between them, and Figure 4.18d shows the IP dislocations from this individual. This individual has a fitness metric of 13.64, an average misorientation of 11.1° , and a dislocation density of $2.54 \times 10^{14} \text{ m}^{-2}$. It is apparent that this individual is a very poor match to the smooth model orientation data, which shows that the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors are necessary in order to produce a good match to the smooth model. Thus the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors are the only Burgers vectors required for, and capable of, producing a good orientation match between the smooth model and the genetic algorithm model. From examination of individuals across multiple runs we have determined that the $[\bar{1}11]$ dislocations are required for the model to match the lower right region of the triangle, and that the $[\bar{1}\bar{1}1]$ Burgers vectors are required for matching the rest of the triangle. The OOP dislocations were not shown because there is no obvious pattern. While they are not random, and they do affect the misorientation, it is difficult to determine if the genetic algorithm model tends to select a similar pattern between runs or whether there are multiple configurations that are similarly effective.

The OOP dislocations were included as a means to relieve strains in the tangential direction (perpendicular to the radial direction, *i.e.* changing θ but keeping r constant in polar

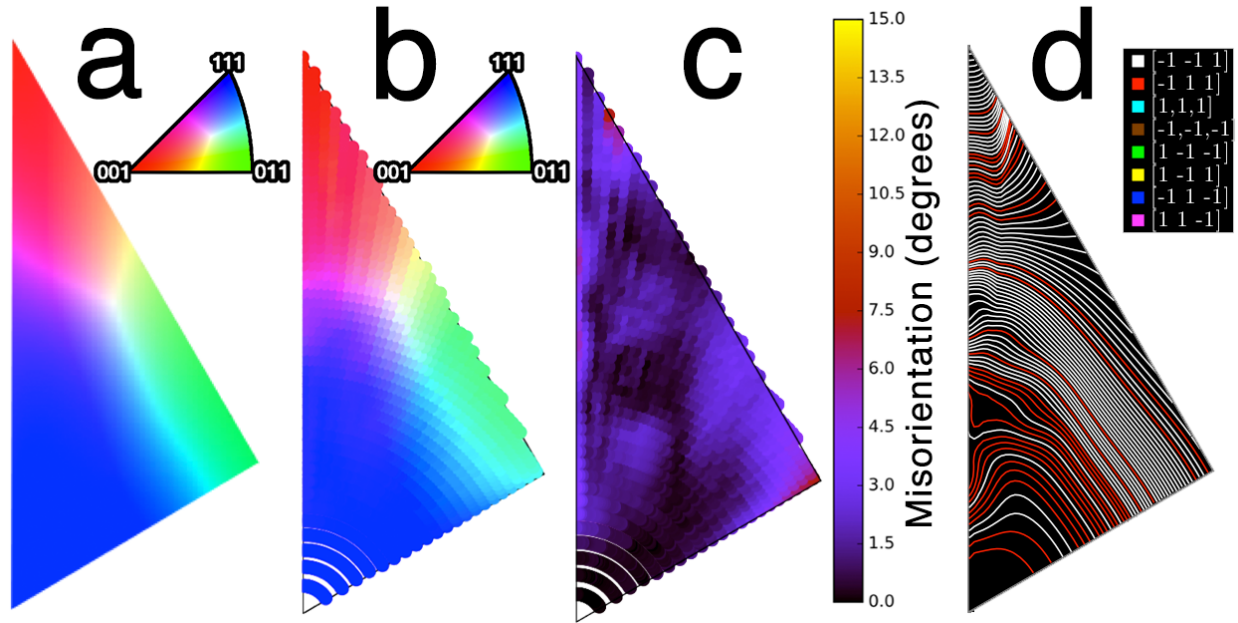


Figure 4.17: Orientation comparison between the smooth model and the genetic algorithm using only $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors. (a) is the orientation data from the smooth model, (b) is orientation data from the genetic algorithm, (c) is the misorientation between (a) and (b), and (c) and (d) are the in-plane and out-of-plane dislocations, respectively, colored by Burgers vector. This individual has a fitness metric of 5.08, an average misorientation of 1.36° , and a dislocation density of $3.72 \times 10^{14} \text{ m}^{-2}$. The low misorientation indicates that the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ are sufficient for the genetic algorithm model to match the orientation data of the smooth model.

coordinates). As the lattice rotates outward in two slightly different radial directions, the misorientation between two adjacent areas eventually becomes too large to be accommodated by elastic strains and a dislocation must be inserted. However, we do not explicitly calculate elastic strains in our model, so existing OOP dislocations may not fully account for misorientations in the tangential direction. If the misorientation for each pixel is computed relative to the previous pixel and divided by the distance between them, this yields what we call the misorientation gradient. Figure 4.19 shows a plot of the misorientation gradient for both the smooth model, in Figure 4.19a, and the individual from Figure 4.14, Figure 4.19b, both colored by their respective scale bars. In Figure 4.19a the misorientation gradient was computed in the x direction and in Figure 4.19b it was computed in the tangential direction, perpendicular to the radial direction. Both Figure 4.19a and Figure 4.19b have their maximum misorientation gradient in the same part of the triangle, but it is not clear why

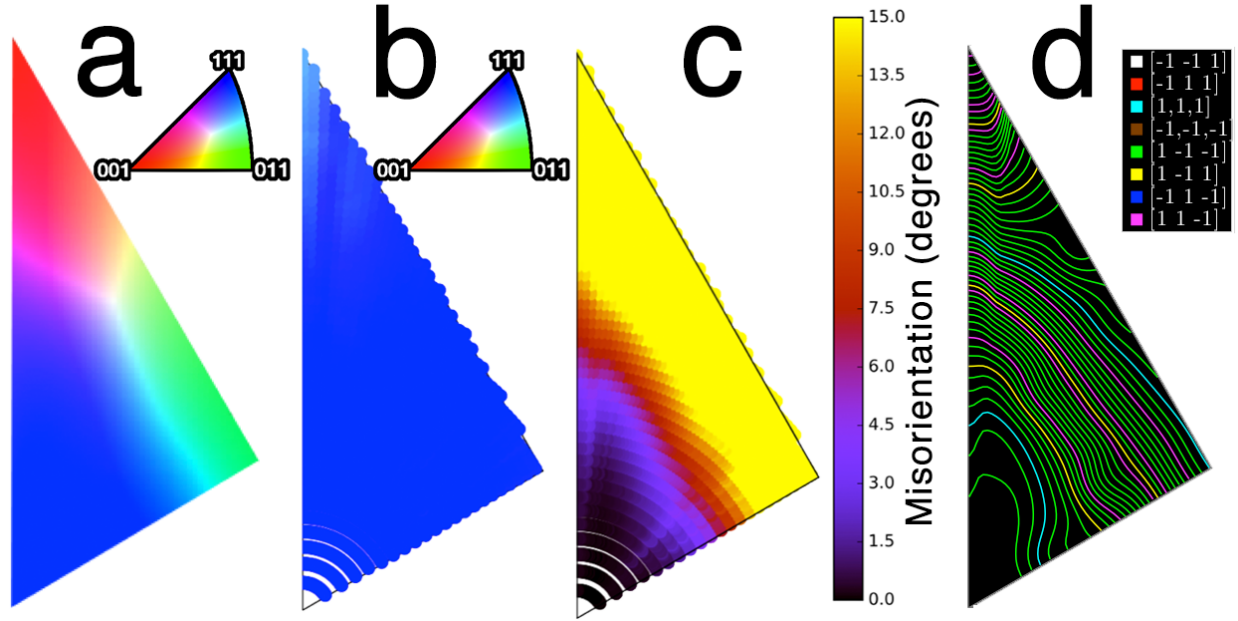


Figure 4.18: Orientation comparison between the smooth model and the genetic algorithm, run using all $\langle 111 \rangle$ -type Burgers vectors except $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$. (a) is the orientation data from the smooth model, (b) is orientation data from the genetic algorithm, (c) is the misorientation between (a) and (b), and (c) and (d) are the in-plane and out-of-plane dislocations, respectively, colored by Burgers vector. This individual has a fitness metric of 13.64, an average misorientation of 11.1° , and a dislocation density of $2.54 \times 10^{14} \text{ m}^{-2}$. The high misorientation indicates that the $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ Burgers vectors are necessary for the genetic algorithm model to match the orientation data of the smooth model.

the genetic algorithm model result, Figure 4.19b, has misorientation gradients that are so much higher than those of the smooth model, Figure 4.19a. While most of Figure 4.19b is below $5^\circ/\mu\text{m}$ near the top of the triangle there are several pixels that are at $20^\circ/\mu\text{m}$, in contrast with Figure 4.19a where the maximum is $3.5^\circ/\mu\text{m}$. This may be because of the spacing of the grid of points at which the orientation was calculated as well as the way dislocations were bundled, so that reducing the bundle size to have more IP dislocations as well as calculating the orientation at more points would reduce the pixel-to-pixel orientation gradient. This would require only minor changes to the model but would take substantially longer to compute. It may also be caused by something more fundamental in the model, such as having separate IP and OOP dislocations, and that allowing dislocations to have both IP and OOP components might reduce the misorientation gradient. However, this would require substantial changes to the model, or possibly even a new model.

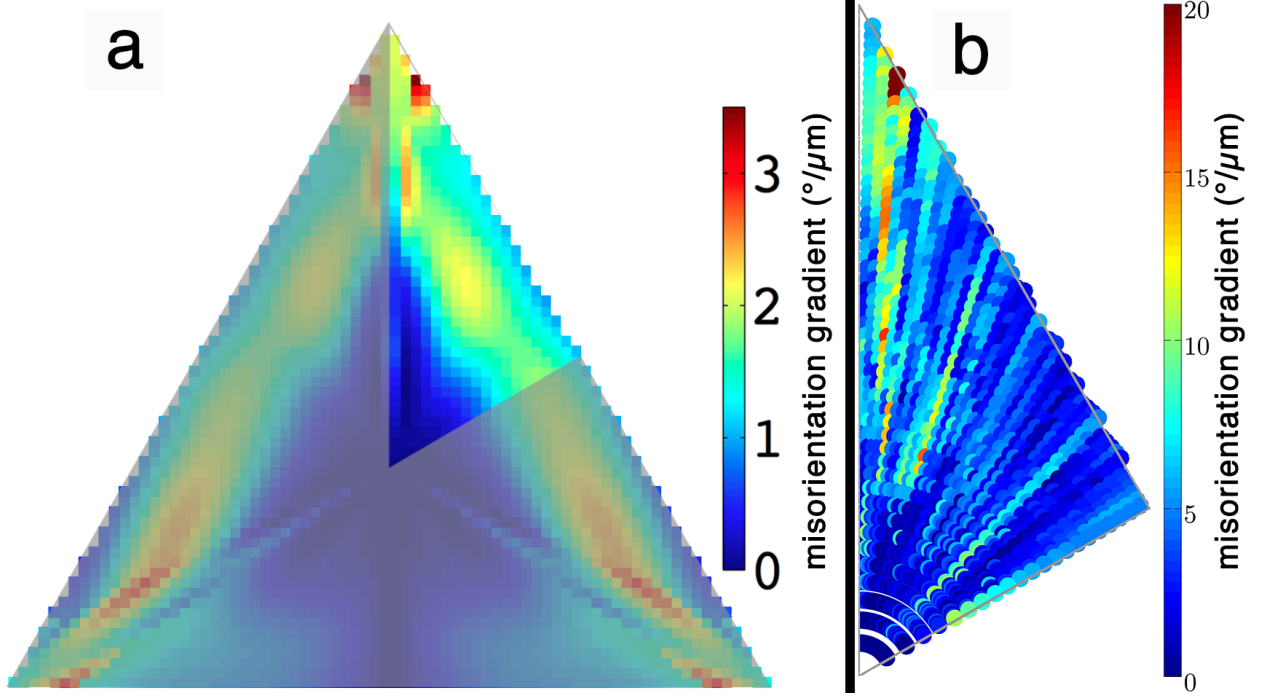


Figure 4.19: A plot of the pixel-to-pixel misorientation gradient for (a) the smooth model and (b) the individual in Figure 4.14, both colored by the same scale bar. This is a plot of the misorientation between adjacent pixels divided by the distance between them. For (a) this is in the x direction and for (b) this is in the tangential direction, perpendicular to the radial direction. The same regions of both triangles have the highest misorientation but it is not clear why the misorientation gradients are so much higher in (b) than in (a).

The dislocation densities generated by this genetic algorithm model are comparable to GND densities in real systems taking into account the differences in orientation gradients relative to their orientation gradients. For example, Kysar *et al.* [42] performed wedge indentation on single-crystal Cu and Al, after which they observed orientation gradients of up to approximately $0.4^\circ/\mu\text{m}$, and GND densities $\sim 10^{13} \text{ m}^{-2}$. The phase-transformed Ta films have orientation gradients approximately an order of magnitude larger, about $4^\circ/\mu\text{m}$, and the dislocation densities generated by the genetic algorithm model are approximately an order of magnitude larger as well, $\sim 10^{14} \text{ m}^{-2}$. Demir *et al.* [59] deformed nickel by equal channel angular pressing and observed orientation gradients slightly larger than those of the phase-transformed films, $\sim 10^\circ/\mu\text{m}$, and calculated GND densities slightly larger than those of the genetic algorithm model, $\sim 10^{15} \text{ m}^{-2}$. A key difference between the phase-transformed films and the work by Kysar *et al.* and Demir *et al.*, however, is that their samples are

heavily deformed whereas the tantalum films are not. It is unusual that our films have such high orientation gradients, as a more typical magnitude for orientation gradients observed in literature is $\sim 0.1^\circ/\mu\text{m}$, whereas our films have orientation gradients of $4^\circ/\mu\text{m}$. It is also unusual that our films have any orientation gradients given that they are not mechanically deformed materials, as nearly all orientation gradients reported in literature are caused by deformation. Only Demir *et al.* [58] and Divinski *et al.* [92] have reported orientation gradients comparable to those of the phase transformed films over a similar distance, but in both cases the orientation gradients are caused by deformation. In Demir *et al.*, the orientation gradients are localized around the indentation site and do not extend past several μm into the sample, and in Divinski *et al.* the gradient is only in one grain, as compared with the phase-transformed tantalum films which are not deformed and have orientation gradients that occur throughout the entire film and are not localized.

The genetic algorithm model was constructed with the assumption that all IP dislocations were in the same plane. This two-dimensional assumption is reasonable given that we only have EBSD orientation data from the surface of the film. However, this microstructure is so different from that of a standard thin film that it is likely that the orientation is not the same throughout the thickness of the film, *i.e.* that there are orientation gradients through the thickness of the film. This means that it may be useful to extend the model from two dimensions to three, and allow the dislocations to no longer be in the same plane. However, this might require essentially a new model and would need substantially more knowledge about the microstructure of the phase-transformed films. It would be necessary to examine orientation data from cross-sections of the film in order to determine whether the orientation is the same on the surface as throughout the thickness of the film or whether there are orientation gradients throughout the thickness of the film as well. This is necessary because accounting for orientation gradients through the thickness of the film would require substantial modifications to the model, or possibly the development of a new model.

This model has been successful where our parametrized Nye tensor approach for cal-

culating GND densities has not partially because this model treats dislocations as discrete and continuous, whereas our Nye analysis only takes into account adjacent pixels with no consideration of how the dislocations in one pixel relate to those in other pixels. This is because the Nye tensor uses a continuum approach to crystal plasticity [35], and therefore provides no information on specific dislocations but instead can only determine generalized GND density. That is, the parametrized Nye tensor approach requires that some particular pixel must have some density of GNDs on each slip system, but there is no relation between those GNDs and those in any other pixel. By contrast, the genetic algorithm model uses specific, individual dislocations, instead of averaging at each pixel. This allows it to give each dislocation a shape, which results in a dislocation structure that is easier to interpret than just regions of high and low GND density.

The prevalence of these two dislocation types, as well as the consistency of their location, suggests they could give rise to the orientation gradient microstructure. Baker *et al.* [32] have proposed that there are two possible mechanisms that may introduce dislocations during the phase transformation. One is that they are created to relieve stresses that arise at the phase boundary during the transformation. The phase transformation causes a large change in stress in the film, with the phase-transformed film at least 1 GPa more tensile than the as-deposited film [24]. This is primarily due to densification of the film, as the α phase is approximately 2.5% denser than the β phase [24]. This means that there would be large shear stresses at the phase boundary, and in order to relax these stresses the α phase may incorporate dislocations. If there were a particular Burgers vector and dislocation character that best reduced the stress, this could lead to arrays of aligned dislocations. As the phase front progressed throughout the film more dislocations would be added to relax stress, resulting in a rotation of the lattice throughout the film. The other potential mechanism that they suggested was that dislocations arise because of kinetic limitations on diffusion during the phase transformation. The difference in density between the α and β phase would result in the formation of a surface step at the phase interface whose height is

approximately an order of magnitude larger than the diffusion length of tantalum, given the thickness of these films [24]. This would result in tantalum atoms at the surface of the film attempting to diffuse downwards from β and attach to the α lattice, but being unable to reach the appropriate location and instead creating extra half-planes in the α phase. As the phase front moved through the film extra half-planes would continue to be added into the α phase, causing the lattice to rotate. This mechanism can only produce edge dislocations.

The computed dislocation structure is consistent with both mechanisms. Our genetic algorithm model includes two types of dislocations, those that lie in the plane and those that are out of the plane. The stress-based mechanism could account for both types of dislocations, while the kinetically limited diffusion-based mechanism could only account for IP dislocations. The stress-based mechanism might be able to produce OOP screw dislocations or IP edge dislocations in order to relieve the stress at the phase boundary. It is also possible that this mechanism could produce mixed dislocations that are neither entirely in or out of the plane of the film. The kinetically limited diffusion-based mechanism could produce dislocations that are consistent with the IP dislocations but not the OOP dislocations in the model. As the phase progressed and β atoms were unable to diffuse fast enough to attach to the α lattice some of the planes of α atoms would not continue as the phase grew, resulting in edge dislocations with Burgers vectors pointing towards up or down through the thickness of the film.

In reality it is unlikely that the dislocations in the film would correspond directly to those in the model, but more work is needed to determine the nature of the dislocations in the film. Specifically, if cross-sections of partially transformed films could be observed in TEM or EBSD with fine spatial resolution, that would yield information about the distribution of dislocations through the thickness of the film. If cross-sections of partially-transformed films showed OOP screw dislocations or mixed dislocations that were partially OOP and partially IP that would suggest that the stress-based mechanism is more likely, and if they showed edge dislocations entirely in the plane with their Burgers vectors pointing towards the origin

of the α phase that would suggest that the kinetically-limited diffusion-based mechanism is more likely. Film cross-sections would also determine if the film has the same orientation throughout its thickness or whether the orientation changes vertically in the film.

4.6 Summary and Conclusions

When tantalum thin films are phase transformed from β to α they exhibit orientation gradients as high as $4^\circ/\mu\text{m}$ over tens of μm . Dislocations must be present in order to account for these orientation gradients, and we want to understand the dislocation structure in these films in order to better understand the phase transformation. However, the only tool previously used in literature to analyze orientation gradients in terms of dislocations, calculating geometrically necessary dislocation density with the Nye tensor, does not work well on our films because of noise in the orientation data. We then made an analytical model that generates an artificial microstructure with smooth orientation gradients and which matches key features of the microstructure of the phase-transformed tantalum films. However the Nye tensor analysis of this model microstructure does not yield useful information because the rotations in the model are not associated with dislocations, so we took the opposite approach and created a genetic algorithm that generates dislocation structures that account for the orientation gradients in the orientation data produced by the smooth analytical model.

The smooth analytical model generates orientation data in a radial manner by rotating outward from a center starting orientation. The close match between the orientation data it produces and EBSD orientation data from the phase-transformed film, validated by OOP orientation maps and pole figures, shows that this radial process can account for observed orientation gradients. Pole figures of the smooth analytical model data also show that the model is able to reproduce twin-like behavior present in the film orientation data. The genetic algorithm model then uses these data to evaluate the dislocation structures it generates by computing what orientations each dislocation structure would cause and then calculating

the misorientation between its orientation data and those of the smooth analytical model. By minimizing this misorientation as well as dislocation density, the genetic algorithm consistently converges on similar dislocation structures with similar misorientation and dislocation density values across multiple runs. The genetic algorithm model also consistently selects two dislocation types, $[\bar{1}\bar{1}1]$ and $[1\bar{1}1]$, which are shown to be necessary for the orientations produced by the dislocation structure to match those of the smooth analytical model. The dislocations in the model have also been linked to proposed mechanisms in the phase transformation that could produce the orientation gradient microstructure.

Chapter 5:

Conclusions and Outlook

We have obtained a dislocation structure that produces an orientation map that closely matches the phase transformed microstructure. In doing so we have developed two models, one that produces orientation data that closely match the microstructure of a phase-transformed film, showing that it is possible to generate this structure in a radial manner. The other model developed was a genetic algorithm that produced dislocation structures capable of accounting for the orientation gradients produced by the first model. The genetic algorithm model consistently generates similar dislocation structures, with most ($\sim 80\%$) $[\bar{1}\bar{1}1]$ and the rest $[\bar{1}11]$, that produce an average misorientation of less than 2° from the first model.

There are two directions that future work could go, theoretical and experimental. For theoretical work, possibilities include examining mechanisms of the phase transformation in more detail to determine how they correspond to the dislocation structure produced by the model, specifically the alignment of burgers vectors; revisiting the parametrized Nye tensor analysis using only the two $[\bar{1}\bar{1}1]$ and $[\bar{1}11]$ slip systems from the model; and performing molecular dynamics simulations to determine how the α phase grows, and how this could affect dislocations in the film. For experimental work, possibilities include examining the film in TEM or high-resolution EBSD, both the surface and in cross-section, for fully and partially phase-transformed films, as a way to determine the actual dislocation structure in the films, how the phase front moves through the film as it transforms, and how this affects the dislocation structure of the films; and investigating other materials for which there have been reported metastable phases similar to tantalum, specifically tungsten [93], chromium [94], and vanadium [95], to see if these materials also develop orientation gradients when phase transformed.

Bibliography

- [1] S. P. Baker, R. A. Knepper, R. S. Fertig III, and K. L. Jackson, “Continuous orientation gradients and discontinuous grain boundaries in phase-transformed α -tantalum thin films.” in process, 2016.
- [2] C. W. Balke, “Ductile Tantalum,” *Industrial & Engineering Chemistry*, vol. 15, pp. 560–562, jun 1923.
- [3] L. Malter and D. B. Langmuir, “Resistance, Emissivities and Melting Point of Tantalum,” *Physical Review*, vol. 55, pp. 743–747, apr 1939.
- [4] M. Oda, A. Ozawa, S. Ohki, and H. Yoshihara, “Ta Film Properties for X-Ray Mask Absorbers,” *Japanese Journal of Applied Physics*, vol. 29, pp. 2616–2619, nov 1990.
- [5] H. J. Mathieu, M. Datta, and D. Landolt, “Thickness of natural oxide films determined by AES and XPS with/without sputtering,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 3, pp. 331–335, mar 1985.
- [6] D. F. Taylor, “Acid Corrosion Resistance of Tantalum, Columbium, Zirconium, and Titanium,” *Industrial & Engineering Chemistry*, vol. 42, pp. 639–639, apr 1950.
- [7] S. Lee, M. Doxbeck, J. Mueller, M. Cipollo, and P. Cote, “Texture, structure and phase transformation in sputter beta tantalum coating,” *Surface and Coatings Technology*, vol. 177-178, pp. 44–51, jan 2004.
- [8] K. Holloway and P. M. Fryer, “Tantalum as a diffusion barrier between copper and silicon,” *Applied Physics Letters*, vol. 57, p. 1736, oct 1990.
- [9] L. A. Clevenger, N. A. Bojarczuk, K. Holloway, J. M. E. Harper, C. Cabral, R. G. Schad, F. Cardone, and L. Stolt, “Comparison of high vacuum and ultra-high-vacuum tantalum diffusion barrier performance against copper penetration,” *Journal of Applied Physics*, vol. 73, p. 300, jan 1993.
- [10] T. Laurila, K. Zeng, J. K. Kivilahti, J. Molarius, and I. Suni, “Effect of oxygen on the reactions in the Si/Ta/Cu metallization system,” *Journal of Materials Research*, vol. 16, pp. 2939–2946, jan 2001.
- [11] J. Fang, T. Hsu, M. Ker, H. Chen, J. Lee, C. Hsu, and L. Yang, “Evaluation of properties of Ta-Ni amorphous thin film for copper metallization in integrated circuits,” *Journal of Physics and Chemistry of Solids*, vol. 69, pp. 430–434, feb 2008.

- [12] P. Baker, "R.F. sputtered tantalum films deposited in an oxygen doped atmosphere," *Thin Solid Films*, vol. 6, pp. R57–R60, nov 1970.
- [13] N. Schwartz, W. Reed, P. Polash, and M. H. Read, "Temperature coefficient of resistance of beta-tantalum films and mixtures with B.C.C.-tantalum," *Thin Solid Films*, vol. 14, pp. 333–346, dec 1972.
- [14] M. H. Read and C. Altman, "A new structure in tantalum thin films," *Applied Physics Letters*, vol. 7, no. 3, pp. 51–52, 1965.
- [15] W. Westwood and F. Livermore, "Phase composition and conductivity of sputtered tantalum," *Thin Solid Films*, vol. 5, pp. 407–420, may 1970.
- [16] A. Arakcheeva, G. Chapuis, and V. Grinevitch, "The self-hosting structure of β -Ta," *Acta Crystallographica Section B Structural Science*, vol. 58, pp. 1–7, jan 2001.
- [17] S. Lee and D. Windover, "Phase, residual stress, and texture in triode-sputtered tantalum coatings on steel," *Surface and Coatings Technology*, vol. 108-109, pp. 65–72, oct 1998.
- [18] M. Nakaishi, M. Yamada, K. Kondo, M. Yamabe, and K. Sugishima, "Anomalous Etching Residues of Sputter-Deposited Ta upon Reactive Ion Etching Using Chlorine-Based Plasmas," *Japanese Journal of Applied Physics*, vol. 31, pp. L1625–L1627, nov 1992.
- [19] K. Kondo, "Stress stabilization of β -tantalum and its crystal structure," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 11, p. 3067, nov 1993.
- [20] T. Yoshihara, "Sputtering of fibrous-structured low-stress Ta films for x-ray masks," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 12, p. 4001, nov 1994.
- [21] A. Jiang, T. A. Tyson, L. Axe, L. Gladczuk, M. Sosnowski, and P. Cote, "The structure and stability of β -Ta thin films," *Thin Solid Films*, vol. 479, pp. 166–173, may 2005.
- [22] L. Feinstein and R. Huttemann, "Factors controlling the structure of sputtered Ta films," *Thin Solid Films*, vol. 16, pp. 129–145, may 1973.
- [23] R. Hoogeveen, M. Moske, H. Geisler, and K. Samwer, "Texture and phase transformation of sputter-deposited metastable Ta films and Ta/Cu multilayers," *Thin Solid Films*, vol. 275, pp. 203–206, apr 1996.

- [24] R. Knepper, B. Stevens, and S. P. Baker, "Effect of oxygen on the thermomechanical behavior of tantalum thin films during the β - α phase transformation," *Journal of Applied Physics*, vol. 100, p. 123508, dec 2006.
- [25] H.-J. Lee, K.-W. Kwon, C. Ryu, and R. Sinclair, "Thermal stability of a Cu/Ta multilayer: an intriguing interfacial reaction," *Acta Materialia*, vol. 47, pp. 3965–3975, nov 1999.
- [26] L. Liu, H. Gong, Y. Wang, J. Wang, A. Wee, and R. Liu, "Annealing effects of tantalum thin films sputtered on [001] silicon substrate," *Materials Science and Engineering: C*, vol. 16, no. 1, pp. 85–89, 2001.
- [27] L. A. Clevenger, A. Mutscheller, J. M. E. Harper, C. Cabral, and K. Barmak, "The relationship between deposition conditions, the beta to alpha phase transformation, and stress relaxation in tantalum thin films," *Journal of Applied Physics*, vol. 72, p. 4918, nov 1992.
- [28] R. Knepper and S. P. Baker, "Coefficient of thermal expansion and biaxial elastic modulus of β phase tantalum thin films," *Applied Physics Letters*, vol. 90, no. 18, p. 181908, 2007.
- [29] C. Cabral, "Repeated compressive stress increase with 400 C thermal cycling in tantalum thin films due to increases in the oxygen content," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 12, p. 2818, jul 1994.
- [30] D. Wu, M. Lee, and T. Lin, "Properties of multilayered thin films for thermal ink-jet printing devices," *Applied Surface Science*, vol. 92, pp. 626–629, feb 1996.
- [31] R. A. Knepper, *Thermomechanical behavior and microstructure evolution of tantalum thin films during the beta-alpha phase transformation*. PhD thesis, Cornell University, jan 2007.
- [32] E. A. Ellis, M. A. Linne, and S. P. Baker, "The Influence of Sputter Pressure on Tantalum Film Microstructure and Phase Transformation." in process, 2016.
- [33] E. A. Ellis, M. Chmielus, and S. P. Baker, "Effect of sputter pressure on Ta thin films: Beta phase formation, texture, and stresses." in process, 2016.
- [34] A. Dirks and H. Leamy, "Columnar microstructure in vapor-deposited thin films," *Thin Solid Films*, vol. 47, pp. 219–233, dec 1977.
- [35] J. F. Nye, "Some geometrical relations in dislocated crystals," *Acta Metallurgica*, vol. 1, pp. 153–162, mar 1953.

- [36] E. Kröner, *Continuum Theory of Dislocation and Self-Stresses*, vol. 5. Verlag, Berlin: Springer, 1958.
- [37] E. Kröner, “Dislocations and continuum mechanics,” *Applied Mechanics Reviews*, vol. 15, no. 8, pp. 599–606, 1962.
- [38] M. F. Ashby, “The deformation of plastically non-homogeneous materials,” *Philosophical Magazine*, vol. 21, no. 170, pp. 399–424, 1970.
- [39] A. Arsenlis and D. Parks, “Crystallographic aspects of geometrically-necessary and statistically-stored dislocation density,” *Acta Materialia*, vol. 47, pp. 1597–1611, mar 1999.
- [40] S. Sun, B. L. Adams, and W. E. King, “Observations of lattice curvature near the interface of a deformed aluminium bicrystal,” *Philosophical Magazine A*, vol. 80, pp. 9–25, jan 2000.
- [41] B. El-Dasher, B. Adams, and A. Rollett, “Viewpoint: experimental recovery of geometrically necessary dislocation density in polycrystals,” *Scripta Materialia*, vol. 48, no. 2, pp. 141–145, 2003.
- [42] J. W. Kysar, Y. X. Gan, T. L. Morse, X. Chen, and M. E. Jones, “High strain gradient plasticity associated with wedge indentation into face-centered cubic single crystals: Geometrically necessary dislocation densities,” *Journal of the Mechanics and Physics of Solids*, vol. 55, pp. 1554–1573, jul 2007.
- [43] R. J. McCabe, A. Misra, and T. E. Mitchell, “Experimentally determined content of a geometrically necessary dislocation boundary in copper,” *Acta Materialia*, vol. 52, no. 3, pp. 705–714, 2004.
- [44] D. Field, P. Trivedi, S. Wright, and M. Kumar, “Analysis of local orientation gradients in deformed single crystals,” *Ultramicroscopy*, vol. 103, no. 1, pp. 33–39, 2005.
- [45] D. P. Field, M. M. Nowell, P. Trivedi, S. I. Wright, and T. Lillo, “Local Orientation Gradient and Recrystallization of Deformed Copper,” *Solid State Phenomena*, vol. 105, pp. 157–162, 2005.
- [46] W. D. Nix, “Elastic and plastic properties of thin films on substrates: nanoindentation techniques,” *Materials Science and Engineering: A*, vol. 234-236, pp. 37–44, aug 1997.
- [47] M. Haque and M. Saif, “Strain gradient effect in nanoscale thin films,” *Acta Materialia*, vol. 51, pp. 3053–3061, jun 2003.

- [48] V. Srikant, J. S. Speck, and D. R. Clarke, “Mosaic structure in epitaxial thin films having large lattice mismatch,” *Journal of Applied Physics*, vol. 82, no. 9, p. 4286, 1997.
- [49] Q. Ma and D. R. Clarke, “Size dependent hardness of silver single crystals,” *Journal of Materials Research*, vol. 10, pp. 853–863, apr 1995.
- [50] W. D. Nix and H. Gao, “Indentation size effects in crystalline materials: A law for strain gradient plasticity,” *Journal of the Mechanics and Physics of Solids*, vol. 46, pp. 411–425, mar 1998.
- [51] H. Gao, Y. Huang, W. Nix, and J. Hutchinson, “Mechanism-based strain gradient plasticity I. Theory,” *Journal of the Mechanics and Physics of Solids*, vol. 47, pp. 1239–1263, apr 1999.
- [52] N. Fleck and J. Hutchinson, “Strain Gradient Plasticity,” in *Advances in Applied Mechanics*, vol. 33, pp. 295–361, 1997.
- [53] N. Fleck, G. Muller, M. Ashby, and J. Hutchinson, “Strain gradient plasticity: Theory and experiment,” *Acta Metallurgica et Materialia*, vol. 42, pp. 475–487, feb 1994.
- [54] M. T. Welsch, M. Henning, M. Marx, and H. Vehoff, “Measuring the Plastic Zone Size by Orientation Gradient Mapping (OGM) and Electron Channeling Contrast Imaging (ECCI),” *Advanced Engineering Materials*, vol. 9, pp. 31–37, feb 2007.
- [55] B. Li, A. Godfrey, and Q. Liu, “Investigation of Macroscopic Grain Sub-Division of an IF-Steel during Cold-Rolling,” *Materials Science Forum*, vol. 408-412, pp. 1185–1190, 2002.
- [56] M. Ferry, “Influence of fine particles on grain coarsening within an orientation gradient,” *Acta Materialia*, vol. 53, pp. 773–783, feb 2005.
- [57] B. Klusemann, B. Svendsen, and H. Vehoff, “Modeling and simulation of deformation behavior, orientation gradient development and heterogeneous hardening in thin sheets with coarse texture,” *International Journal of Plasticity*, vol. 50, pp. 109–126, nov 2013.
- [58] E. Demir, D. Raabe, N. Zaafarani, and S. Zaefferer, “Investigation of the indentation size effect through the measurement of the geometrically necessary dislocations beneath small indents of different depths using EBSD tomography,” *Acta Materialia*, vol. 57, pp. 559–569, jan 2009.
- [59] M. Calcagnotto, D. Ponge, E. Demir, and D. Raabe, “Orientation gradients and geometrically necessary dislocations in ultrafine grained dual-phase steels studied by 2D

- and 3D EBSD,” *Materials Science and Engineering: A*, vol. 527, pp. 2738–2746, apr 2010.
- [60] X. Maeder, C. Niederberger, S. Christiansen, A. Bochmann, G. Andrä, A. Gawlik, F. Falk, and J. Michler, “Microstructure and lattice bending in polycrystalline laser-crystallized silicon thin films for photovoltaic applications,” *Thin Solid Films*, vol. 519, pp. 58–63, oct 2010.
 - [61] M. Phillips, R. Spolenak, N. Tamura, W. Brown, A. MacDowell, R. Celestre, H. Padmore, B. Batterman, E. Arzt, and J. Patel, “X-ray microdiffraction: local stress distributions in polycrystalline and epitaxial thin films,” *Microelectronic Engineering*, vol. 75, no. 1, pp. 117–126, 2004.
 - [62] W. Heinz and G. Dehm, “Grain resolved orientation changes and texture evolution in a thermally strained Al film on Si substrate,” *Surface and Coatings Technology*, vol. 206, pp. 1850–1854, dec 2011.
 - [63] A. Bastos, S. Zaefferer, and D. Raabe, “Three-dimensional EBSD study on the relationship between triple junctions and columnar grains in electrodeposited Co-Ni films,” *Journal of microscopy*, vol. 230, pp. 487–98, jun 2008.
 - [64] D. Raabe, Z. Zhao, S.-J. Park, and F. Roters, “Theory of orientation gradients in plastically strained crystals,” *Acta Materialia*, vol. 50, pp. 421–440, jan 2002.
 - [65] A. M. Turing, “Computing Machinery and Intelligence,” *Mind*, vol. LIX, no. 236, pp. 433–460, 1950.
 - [66] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, 2005.
 - [67] D. B. Fogel, *Evolutionary Computation: The Fossil Record*. Wiley-IEEE Press, jun 1998.
 - [68] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., oct 1989.
 - [69] J. H. Holland, *Adaptation in Natural and Artificial Systems: an introductory analysis with applications to biology, control, and artificial intelligence*. Ann Arbor, Michigan: University of Michigan Press, 1975.
 - [70] K. Wloch and P. J. Bentley, “Optimising the Performance of a Formula One Car Using a Genetic Algorithm,” in *International Conference on Parallel Problem Solving from Nature* (X. Yao, E. K. Burke, J. A. Lozano, J. Smith, J. J. Merelo-Guervós, J. A. Bullinaria,

- J. E. Rowe, P. Tio, A. Kabán, and H.-P. Schwefel, eds.), vol. 3242 of *Lecture Notes in Computer Science*, pp. 702–711, Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.
- [71] G. Hornby, A. Globus, D. Linden, and J. Lohn, “Automated Antenna Design with Evolutionary Algorithms,” *American Institute of Aeronautics and Astronautics*, pp. 19–21, 2006.
 - [72] D. Deaven and K. Ho, “Molecular geometry optimization with a genetic algorithm,” *Physical review letters*, vol. 75, pp. 288–291, jul 1995.
 - [73] J. A. Wright, H. A. Loosemore, and R. Farmani, “Optimization of building thermal design and control by multi-criterion genetic algorithm,” *Energy and Buildings*, vol. 34, pp. 959–972, oct 2002.
 - [74] M. M. Eusuff and K. E. Lansey, “Optimization of Water Distribution Network Design Using the Shuffled Frog Leaping Algorithm,” *Journal of Water Resources Planning and Management*, vol. 129, pp. 210–225, may 2003.
 - [75] Y. Rahmat-Samii and E. Michielssen, *Electromagnetic Optimization by Genetic Algorithms*, vol. 42. New York, NY: John Wiley & Sons, Inc., 1999.
 - [76] S. E. Restrepo, S. T. Giraldo, and B. J. Thijsse, “A genetic algorithm for generating grain boundaries,” *Modelling and Simulation in Materials Science and Engineering*, vol. 21, p. 055017, jul 2013.
 - [77] A. L.-S. Chua, N. A. Benedek, L. Chen, M. W. Finnis, and A. P. Sutton, “A genetic algorithm for predicting the structures of interfaces in multicomponent systems,” *Nature materials*, vol. 9, pp. 418–22, may 2010.
 - [78] J. Zhang, C.-Z. Wang, and K.-M. Ho, “Finding the low-energy structures of Si[001] symmetric tilted grain boundaries with a genetic algorithm,” *Physical Review B*, vol. 80, p. 174102, nov 2009.
 - [79] T. Murata, H. Ishibuchi, and H. Tanaka, “Multi-objective genetic algorithm and its applications to flowshop scheduling,” *Computers & Industrial Engineering*, vol. 30, pp. 957–968, sep 1996.
 - [80] G. C. Wang and T. M. Lu, *RHEED transmission mode and pole figures: Thin film and nanostructure texture analysis*, vol. 9781461492. Springer New York, 2014.
 - [81] F. Bachmann, R. Hielscher, and H. Schaeben, “Texture Analysis with MTEX Free and Open Source Software Toolbox,” *Solid State Phenomena*, vol. 160, pp. 63–68, feb 2010.

- [82] M. H. A. Nawaz and B. L. Mordike, “Slip Geometry of Tantalum and Tantalum Alloys,” *Physica Status Solidi (a)*, vol. 32, pp. 449–458, dec 1975.
- [83] D. W. Matson, “High rate sputter deposition of wear resistant tantalum coatings,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 10, p. 1791, jul 1992.
- [84] W. J. Kitchingman, “The atomic mechanism of the body-centred cubic to σ -phase transformation,” *Acta Crystallographica Section A*, vol. 24, pp. 282–286, mar 1968.
- [85] F.-A. Fortin, F.-M. De Rainville, M.-A. G. Gardner, M. Parizeau, and C. Gagné, “DEAP: evolutionary algorithms made easy,” *The Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jan 2012.
- [86] Y. Hold-Geoffroy, O. Gagnon, and M. Parizeau, “Once you SCOOP, no need to fork,” in *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment - XSEDE '14*, (New York, New York, USA), pp. 1–8, ACM Press, jul 2014.
- [87] S. van der Walt, S. C. Colbert, and G. Varoquaux, “The NumPy Array: A Structure for Efficient Numerical Computation,” *Computing in Science & Engineering*, vol. 13, pp. 22–30, mar 2011.
- [88] E. Jones, T. Oliphant, and P. Peterson, “SciPy: Open Source Scientific Tools for Python,” 2001.
- [89] J. D. Hunter, “Matplotlib: A 2D Graphics Environment,” *Computing in Science & Engineering*, vol. 9, pp. 90–95, may 2007.
- [90] S. Osher and R. P. Fedkiw, “Level Set Methods: An Overview and Some Recent Results,” *Journal of Computational Physics*, vol. 169, pp. 463–502, may 2001.
- [91] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, apr 2002.
- [92] S. V. Divinski, G. Reglitz, M. Wegner, M. Peterlechner, and G. Wilde, “Effect of pinning by an orientation gradient on the thermal stability of ultrafine grained Ni produced by equal channel angular pressing,” *Journal of Applied Physics*, vol. 115, pp. 113–503, mar 2014.
- [93] S. M. Rossnagel, I. C. Noyan, and C. Cabral, “Phase transformation of thin sputter-

deposited tungsten films at room temperature,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 20, no. 5, p. 2047, 2002.

- [94] M. O’Keefe, S. Horiuchi, J. Rigsbee, and J. Chu, “Effect of oxygen and carbon on the formation and stability of A-15 crystal structure chromium thin films,” *Thin Solid Films*, vol. 247, pp. 169–177, jul 1994.
- [95] Y. Tian, F. Jona, and P. Marcus, “Metastable phase of vanadium,” *Physical Review B*, vol. 58, pp. 14051–14055, nov 1998.